Contents

A Formal Approach to the Design of Construction Information Management Systems

Osama Y. Abudayyeh [1] and William J. Rasdorf [2]

ABSTRACT

Effective management of construction projects depends on good access to and control of data, especially data pertaining to cost and schedule control functions. Long recognizing the need to integrate these interrelated functions, researchers have proposed conceptual models to achieve this purpose. However, the proper design of an automated solution has been lacking that would support the needs of integrated cost and schedule control and would provide distributed access to data for different processing needs. The purpose of this paper is to show how such an automated solution (represented by a data storage model) was properly designed and developed. The data storage model is developed using the first three steps of the process used to model engineering problems. The first step, problem definition, was accomplished by using the work packaging model to solve the integration problem and by using data collection forms developed by R.S. Means Company to identify what data items are collected in cost and schedule control. The second step, conceptual modeling, utilizes Nijssen's Information Analysis Methodology (NIAM) for data modeling. For the third step, computational modeling, the relational data model is selected because of its widely accepted mechanisms for storing, retrieving, and using data, and for its concepts (tables and rows) that lend themselves well to representing and processing cost and schedule data. The Optimal Normal Forms (ONF) algorithm is used for mapping from the conceptual to the computational model. Each of the above steps is described in detail in this paper.

1 Introduction

Information plays a key role in construction project management. In order for a construction project to be well managed, data from past projects, stored in a historical database, as well as data from the project at hand, must be readily available. It is an essential and valuable resource for project planning, control, reporting, and decision-making tasks. In each of these tasks, effective management of information is an integral part of a successful project management system, whose primary objective is completing the project on time and within budget limitations while meeting established quality requirements and other specifications.

In the subsections below, the needs for integrating cost and schedule control functions and for automating the methods of data storage and use pertaining to these functions are discussed. The process of modeling engineering problems will also be reviewed and the development of an automated information management system will be demonstrated.

[1] Graduate Assistant, Department of Civil Engineering, North Carolina State University, Box 7908, Raleigh, NC 27695

[2] Associate Professor, Department of Civil Engineering, North Carolina State University, Box 7908, Raleigh, NC 27695

1.1 Problems and Needs of Cost and Schedule Control Functions

A majority of construction projects employ some method of cost and schedule control. Still, many projects suffer from ineffective control due to inefficient flow of information [Kratt 89]. Problems stemming from poor information flow are observed during the projects themselves at the time of their construction, as well as in the corporate historical database. A project may suffer from delays and over-expenditures due to the feedback problems. Historical databases may contain bad or unrealistic data that affects future planning abilities and results in lost bids or reduced profits on future projects. If new goals and plans are based on incorrect or unrealistic historical data, new projects will suffer from a continuous cycle of poor performance: a submitted bid may exceed what is really needed, resulting in a lost opportunity to win a contract. Also, an opportunity may be lost to achieve higher profits even if the project is finished on time and within budget, and the project may suffer from delays and cost overruns.

A solution to the problems described above imposes the challenge of integrating cost and schedule control functions. However, construction projects exhibit complex interrelationships between schedules and costs [Hendrickson 89]. A schedule delay has a direct impact on costs. Also, monetary constraints directly affect the duration of a project. Time-cost trade-offs are basic decisions exercised in project management. Although the interdependency between schedule and cost is obvious, it is rare to find project control systems that integrate cost and schedule control functions [Hendrickson 89]. Rather, they remain two separate functions that are performed independently from each other and that use two different control structures, the Work Breakdown Structure (WBS) and the Cost Breakdown Structure (CBS). The WBS is a breakdown of the project into significant units of work starting with the entire project at the highest level on the hierarchy and ending with the tasks needed to complete the project as the lowest level. On the other hand, the CBS is a breakdown of the project into smaller cost centers that contain all classes and categories of items pertaining to the different types of work. Referring to the difference between these two structures, Ibbs, et al., states the following:

> The problem is that the schedule work breakdown structure is defined much differently than the cost breakdown structure.... We schedule the construction of an east wall separately from the west wall, but aggregate their costs [Ibbs 87: Page 108].

It is clear, then, that cost and schedule control functions need to be integrated. Commenting on this problem, Teicholz states that

> Cost control systems are widely used within the construction industry.... These systems are sometimes well integrated with accounting systems, but rarely integrated with scheduling and material management.... There is an obvious need to integrate the cost, schedule, and material management systems so that they reflect a consistent view of the project. In practice, this is a difficult and often neglected task [Teicholz 87: Pages 47 and 57].

However, the difficulty of integrating both functions lies in the level of detail that each function uses, as opposed to the level of detail needed. The cost control function, represented by the CBS, is performed at a less detailed breakdown level than that of the schedule control function, represented by the WBS. For example, a cost account on the CBS may accumulate data relative to man-hour, concrete material, and equipment costs associated with formwork for walls as shown in Figure 1. From a scheduling perspective (WBS), formwork for walls is broken down into several tasks and subtasks (activities). As shown in Figure 1, these may include fabricating formwork of walls in area A for floor 4, erecting formwork of walls in area A for floor 4, stripping formwork of walls in area A for floor 4, and similar tasks for areas B and C of floor 4 and other floors as well. Clearly, the lowest levels of detail can be very different.

The different levels of detail used by each function create a fundamental difference in the way cost and schedule data is presently maintained. Each collection of data has traditionally become independent from the other and is usually maintained separately. Project managers must then integrate the information coming from two sources, which reduces the efficiency of obtaining meaningful information and making prompt and fully informed decisions. For example, project managers might spend time and effort trying to isolate the costs of the concrete work on walls in area A on floor 4 from the total costs accumulated in the walls cost account.

This paper, then, addresses a key issue in achieving effective project control: the integration of cost and schedule control functions. To achieve the desired integration, cost and time data must be acquired and maintained using an established common denominator control account defined using a sufficiently detailed level. However, it is not enough to simply establish and maintain a common denominator. What is also needed are automated methods for acquiring and storing data that supports this concept and level of detail. In other words, a computing-based information management model is needed that supports the requirements of integrated cost and schedule control and provides distributed access to data for different processing needs.

1.2 Modeling Construction Cost and Schedule Data

To develop an automated and integrated solution for the cost and schedule control problem, a four-step process, detailed below, for modeling engineering problems has been used. The process is shown in Figure 2.

The first step in the modeling effort, problem definition, involves identifying the data items needed and describing the behavior of the engineering system with respect to the methods and mechanisms used in acquiring, storing, and processing data. This step was accomplished by using the work packaging model as a solution to the integration problem and also by using data collection forms developed by R.S. Means Company [Abudayyeh 90, Rasdorf 90b, Means 86]. The work packaging model describes the conceptual organization of data as well as the data processing mechanisms involved in integrated cost and schedule control, but it had to be modeled to provide the necessary automated support as shown in Figure 3. In this figure, the work packaging model represents a portion of the first step in the modeling process, where the behavior of the model is formally described. What is missing from the problem definition is the identification of the data items that are used in construction cost and schedule control. An effective control system should typically acquire a large number of data items on a daily basis that covers all aspects of a construction project. To identify what data items are being acquired in cost and schedule control functions, a subset of data collection forms, developed by R.S. Means Company and relevant to both functions, was selected [Means 86]. Utilizing the data-item contents of these forms and the knowledge provided by the work packaging model, the problem definition becomes complete. The specifications pertaining to the relevant knowledge and data items are formulated as a set of examples known by *elementary facts* as will be seen in Sections 3 and 4.

The second step, conceptual modeling, is a formal representation of the problem formalized by step one. It accepts an engineering problem definition as input and produces a conceptual data model that represents the design of the system. In this step, a fairly recent formal data modeling methodology: Nijssen Information Analysis Methodology (NIAM) was utilized [Rasdorf 90a, Nijssen 89]. This methodology helps in the design of the information management system used to support the work packaging model. Specifically, it systematically develops a conceptual data model that captures the meanings of and relationships between the data items as provided by the problem definition. The methodology produces a conceptual data representation model called the Conceptual Schema Diagram (CSD).

The third step, computational modeling, takes a conceptual data model from step two as input and transforms it to a computational model suitable for developing automated environments. For this step, the relational data model was selected. It is believed that the relational data model provides the necessary, standardized, and widely accepted automated mechanisms for data storage and retrieval, and its concepts (tables and rows) are well suited to presenting and processing cost and schedule data. The NIAM methodology provides an algorithm called the Optimal Normal Form (ONF) that transforms a NIAM model into a fifth normal form relational data model. The ONF algorithm was used to transform the CSD of step two into a step three relational database schema.

The fourth and final step of the modeling process, computer modeling, takes the computational model as input and maps it onto a computer data model that represents the automated computer system solution to the problem. In this step, the relational data model can be mapped onto any relational database management system (DBMS) schema using the data definition language (DDL) provided by the DBMS package. This paper, however, focuses only on the first three steps of the engineering-problem modeling process, which represent the formal and systematic approach to conceptually designing an automated information management system, and does not address the computer implementation of the system that constitutes step four.

1.3 Information Engineering

The previous subsection suggested using a formal methodology for the conceptual modeling step of the engineering modeling process. The outcome of NIAM is a CSD that is viewed as an engineering drawing, which requires the approval of the client (owner) before proceeding with the third step of the engineering modeling process [Rasdorf 90a]. This view makes the design of an information management system similar in concept to the design of an engineering system such as a building project. In the latter, the idea of a building becomes a reality after the architectural drawings are developed. The owner then reviews the drawings and approves them before the actual design proceeds. In this study, the architectural design is the conceptual modeling step, the detailed design is analogous to the computational modeling step, and the construction of a building resembles the computer modeling step.

NIAM diagrams have been used by Sandia National Laboratories as a communication vehicle between the designer of an information management system and the user of the information management system, where the actual implementation (computational and computer modeling) of the information system would not proceed without the approval of the client [Sharp 90]. They are used as a set of engineering drawings themselves, to define the design of the structure of the information management system. Once the NIAM diagrams are approved, the computational modeling is initiated followed by the computer modeling.

The organization of this paper generally reflects the first three steps of the engineering modeling process described in this section. Section 2 describes the work packaging model and Means data collection forms. Section 3 provides an overview of the NIAM methodology. Section 4 develops the

NIAM conceptual schema for the work packaging model. Section 5 transforms the NIAM model to a relational data model. Section 6 provides conclusions.

2 The Construction Management Problem Definition

This section describes the work packaging and the Means Company data collection forms that are important to the problem definition step of the engineering-problem modeling process. This section provides the background knowledge pertinent to the work packaging model and identifies the data items that are collected by cost and schedule control functions. The complete set of elementary facts that models this knowledge and data items is provided in Section 4.

2.1 Work Packaging Requirements

The work packaging model was developed to create a unified view of project data, specifically time and cost data, so that the analysis and decision-making processes would be easier to perform. The use of the work packaging model imposes a change in the planning and budgeting philosophy. It also requires a new approach to defining and identifying control accounts and requires designing coding schemes for identifying the different resources at a construction job site. These issues are addressed in the subsections below.

2.1.1 Planning and Budgeting

Planning and budgeting in the work packaging context refer to the scheduling of all authorized work and assigning budgets to manageable units of work (work packages) [DOE 80]. The work packaging model requires the use of the WBS early in the project life cycle. This means that work must be scheduled in accordance with the WBS, and budgets of resource consumption and costs must be prepared for each control account on the WBS.

Because of the change in the planning and budgeting philosophy and because of the cost and schedule control functions integration needs, a unified data organization is imposed on each control account. Such an organization must provide for each control account a scope of work, an account code number, planned start and end dates, a budget for resource consumption, actual start and end dates, and a record of actual resource consumption.

2.1.2 Defining Control Accounts

Details pertaining to the thousands of operations involved in medium to large construction projects are captured by the WBS and also by the the contractor Organization Breakdown Structure (OBS), shown in Figure 4. The WBS shown in the figure represents a small portion (approximately 10%) of a complete WBS example. As shown in Figure 4, organizational functional responsibilities are assigned to each work package. This is accomplished by creating a two-dimensional matrix with one dimension being the WBS element and the other dimension being the OBS element, as is illustrated by Figure 4. An example control account, at the intersection of the concrete work for walls element on the WBS and the concrete foreman element on the OBS, is enlarged at the bottom of the figure. In this control account, the concrete foreman is responsible for the concrete work on area A's walls on floor 4.

2.1.3 Identification System

An identification system is needed to support the work packaging model. It must be designed to provide an easy addressing mechanism to each work package. This means that a coding scheme must be able to identify all of the processes and resources of a construction project. Therefore, to be able to satisfy the addressing mechanism needs, six components are needed in the coding scheme: a WBS, an OBS, a resource, a workers' identification, an activity network identification, and a task coding schemes. These components should be developed and standardized within a company, and hopefully within the construction industry as suggested by the Construction Industry Institute [CII 88, CII 87].

******* I REALLY DO NOT THINK WE SHOULD MENTION THIS INHERE ******** Standardization is an important issue, specifically when information exchange between the different organizations involved during the construction process such as the owner's representatives, architects, designers, contractor(s), and vendors, is of concern. Not only this, but also if information exchange is to take place between the design and construction phases. Some, standardization efforts have taken place in the past such as the Uniform Construction Index (UCI) coding scheme for cost control developed by the Construction Specifications Institute (CSI) [Barrie 84, Halpin 85]. Unfortunately, such standards have been developed for non-integrated cost and schedule control environments, and thus cannot be used in integrated approaches.

The first three code components are packaged in the following generic code: WWWWWW–OOOO–RRR [Halpin 85, Halpin 87]. The first six "W" spaces are used to represent the six-level WBS dimension shown in Figure 4, where each WWWWWW code represents a node on the WBS. The next four "O" spaces are used to represent the elements of the four-level OBS dimension shown in Figure 4, where each OOOO code represents a node on the OBS. The last three "R" spaces are used to represent the three resource types (workforce, material, and equipment), where each RRR code represents a specific resource such as concrete material, concrete finisher, or concrete pump.

The remaining three code components identify other important project information. The fourth code component is the identification number (ID) for each worker on the job site. To satisfy this need, social security numbers can be used. A fifth code component is needed for identifying the network activities after the logical sequence of tasks has been established. This component is most often dictated by the scheduling software package used for performing network computations. A sixth code component is needed to identify the different tasks and their types. This component, suggested by the CII, can be developed within a company for all the tasks defined on the WBS [CII 88, CII 87]. This component is specifically useful in retrieving information from historical databases.

2.2 Construction Data Forms

Presently, construction data is acquired using a variety of forms. A number of data collection forms are available in the literature and in construction firms [Adrian 86, Means 86, Halpin 85, Carlsen 78]. These forms are designed to collect fundamentally the same data items, though in different formats and structures. Because of their extensive and detailed data items content, the R.S. Means Company forms have been selected for identifying what data is currently being collected by the cost and schedule control functions [Abudayyeh 90, Rasdorf 90b].

The R.S. Means Company has developed a large set of construction forms. A subset of these forms was identified as being used to acquire cost and schedule data, which includes the Daily Time

Sheet, Weekly Time Sheet, Daily Construction Report, Labor Cost Record, Job Progress Report, and Project Schedule [Means 86]. Below is a description of each form in this subset.

- Daily Time Sheet: The purpose of this form is to keep a daily record of the activities performed by all workers at a construction job site. It records the daily regular and overtime hours worked and units produced by each worker on the different tasks.
- Weekly Time Sheet: The purpose of this form is to keep a weekly record of one worker's activities at a construction job site. It can be used to record the weekly hours of a worker working on different jobs.
- Daily Construction Report: The purpose of this form is to keep a detailed record of all activities at a construction job site. It records the number and type of workers associated with each cost account, the total hours spent on the account, the material and equipment usage, and other miscellaneous activities such as verbal instructions and site visitors.
- Labor Cost Record: The purpose of this form is to keep a daily and weekly records of total labor costs by trade in each cost account for a construction job site.
- Job Progress Report: The purpose of this form is to keep a periodic (weekly, monthly, etc.) record of job progress (work done or percent complete).
- Project Schedule: The purpose of this form is to keep a graphical record of the start and finish dates of activities, as well as to plan future activities. The form uses the bar chart format. The activities on the bar chart can have a finished, in-progress, or not started status.

This subset of Means forms was analyzed to determine the unique subset of data items that are acquired in support of cost and schedule control functions. The analysis yielded a large number of data items that are currently being collected on a construction job site. Many redundancies were observed in these forms. However, this was to be expected because the two functions are not integrated: some forms are designed to support cost control, while others are designed to separately support schedule control.

The five major categories of data items that were acquired from the unique subset of the Means forms are the following: general data, direct labor hours and labor costs, direct material quantities and material costs, direct equipment hours and equipment costs, and time data. A brief description is provided below and a more detailed analysis of both the Means forms and the data they archive can be found in references [Abudayyeh 90, Rasdorf 90b].

- <u>General Data Items.</u>
  This category includes such data as the project's name, code number, and location; the weather condition on the date of filling the form; and the name of the person filling the form.
- <u>Direct Labor Hours and Labor Costs.</u>
  This category includes such data as cost account code, worker's name and ID, regular and overtime hourly pay rates, and regular and overtime hours expended on different work items by each worker and by each craft.
- <u>Direct Material Quantities and Material Costs.</u>
  This category includes such data as total quantities of direct materials (e.g. concrete , rebars, etc.) used on different work items and charged to different cost accounts, unit costs of the materials, and a description of each material.

- Direct Equipment Hours and Equipment Costs.
  This category includes such data as equipment code, equipment description, hourly rental rate, and hours of operation on each work item.

- Task Time Data.
  This category includes such data as the task code, task description, and actual start and finish dates.

In the cost data items, only those items that contribute to the direct cost of a project are considered. Indirect cost data items, such as overhead and indirect materials, are not addressed in this paper.

3 Conceptual Modeling Using NIAM

The second step in developing an automated solution for an engineering problem, conceptual modeling, is addressed in this section, which serves as an introduction to NIAM. This section will provide a brief overview of its development and will then discuss the two main stages of the NIAM data modeling methodology. These two stages are defining facts and defining constraints. The information on NIAM in this section serves as the foundation for Section 4, which applies the NIAM methodology to modeling the information extracted from the Means forms and the work packaging model.

3.1 Overview of NIAM

The NIAM modeling methodology, first researched by Nijssen and Falkenberg, has been revised into its present form by Halpin [Nijssen 89]. It is a simple, natural approach to semantic modeling which accepts data and constraints and produces a conceptual database schema design that is independent of and can be mapped to any database model. Semantic modeling involves identifying a set of semantic concepts that can be used to formally model the real world and to represent the meanings of data. A semantic model is composed of objects and the relationships between them. The following subsections describe the concepts and principles of NIAM.

3.2 Defining Facts

The first stage of the design procedure, defining facts, starts with examining the inputs to the information system under consideration. Such inputs are normally available in the form of reports, documents, input forms, graphs, etc. These types of inputs are used to represent the meaning and organization of data items pertinent in a given domain. Then, the elementary facts are formally represented using a NIAM formalism (language) to arrive at a preliminary conceptual data model. Finally, the data model is refined to complete the first stage of the CSDP. These issues are further discussed in the following subsections.

3.2.1 Developing Elementary Facts

The first step in defining facts to develop elementary facts. These originate from "representative examples." An elementary fact is defined as a fact that cannot be split into smaller facts while preserving the original information. In developing these facts one should work with a representative and significant set of examples (facts). A set of examples is said to be significant if it provides all the relevant information and constraints about the domain to be modeled [Rasdorf 90a].

To further understand NIAM consider the concepts identified in this paragraph. In NIAM, the *domain of interest*, sometimes referred to as the universe of discourse (UoD), is thought of as a set of entities that define a relationship. A fact type in the UoD asserts that certain entities play certain roles in a relationship. Each entity has a *type* which defines its set of all possible instances. An instance of an entity type in a fact type is called a *label* and the unit of measurement or the reference base for the entity type is called *reference mode*. Each entity type must play at least one *role*. A role is a part played by an entity type in some relationship. Each role is played by exactly one entity type. A role name should be unique within the context of a fact type. Each fact type has an *arity* which indicates the number of entity types involved.

Using a short hand notation for representing fact types, entity types are written with the first character of the name capitalized, reference modes are enclosed in parentheses, text labels are enclosed in single quotes, numeric labels are written as numbers, and role names are italicized. Additionally, the fact type name is written in bold face, followed by a colon, and precedes the elementary fact instance. Examples of a binary (arity of two) and ternary (arity of three) fact types are:

- **WP-Level-Of-Detail:** WBS-Node (code) 134123 *has* Level-Of-Detail (number) 6.
- **Material Budgets:** Control-Account (code) 134123 *has-budget* Quantity (number) 100 *for* Material (code) 100.

3.2.2 NIAM Graphical Language

After elementary facts have been extracted from examples, they need to be represented using a standard and formal method (or language). NIAM provides a formal graphical language to represent facts and roles [Rasdorf 90a, Nijssen 89]. Specifically, in this language, the following symbols are used:

- An ellipse represents an entity type with the name of the entity written inside it and the reference mode written underneath the name enclosed in parentheses.
- A rectangle represents a role.
- A line segment connects an entity type to each role that it plays.
- A contiguous sequence of n role rectangles, each of which is connected to exactly one entity type, represents an n-ary fact type.
- The roles are written as a single predicate having empty gaps indicated by '...' written inside an end rectangle in an n-ary fact types. For roles, the notation indicates the first entity type occupying the first gap in the predicate. The remaining gaps are occupied by the remainder of the entity types in the fact types as they appear from this end to the other end.

The graphical representation of a fact is called a Conceptual Schema Diagram (CSD). Figure 5 shows the CSD for the binary and the ternary fact types enumerated in Section 3.2.1.

3.2.3 Refining the Conceptual Schema

Refining the conceptual schema involves eliminating unnecessary fact types or unnecessary entity types or both. Unnecessary fact types are observed when a fact type can be derived from another fact type(s). Unnecessary entity types occur when two different entity types can be combined into one. One good indicator that two entity types may be combined occurs when both have the same reference mode base [Rasdorf 90a, Nijssen 89]. However, as will be discussed in Section 4.3, not all entity types with the same reference mode need to be combined. This decision depends on the entity types and their respective pieces of information that they model.

In some situations, it is preferred to use *nesting* as an alternative to using a flattened version (a fact type with no nesting). Nesting is a mechanism provided by NIAM that treats a relationship between entity types as an entity type itself, and this entity type is called an *objectified relationship type*. Just as a relationship between entity types is composed of roles, so is the objectified relationship type. A fact type that includes an objectified relationship type is called a *nested fact type*. The objectified relationship type, just like any other entity type, must play at least one role and is represented in NIAM by an ellipse that encloses roles. But how can one decide whether to use a nested version or a flattened one? The answer to this question lies in the following general rule: a nested fact type is used whenever two fact types share all their entity types and the uniqueness constraint (discussed in Section 3.2.1) is applied to the same roles in both fact types; otherwise, the flattened version is preferred. A detailed discussion of the nesting mechanism can be found in references [Rasdorf 90a, Nijssen 89].

3.3 Defining Constraints

After the conceptual schema diagram has been developed, the next stage in the CSDP is to represent the constraints that govern the behavior of the elementary fact types and entity types on the diagram. Such constraints play a key role when the conceptual schema is mapped onto a relational data model. In this section three constraint types will be described: uniqueness, entity type, and mandatory and optional roles. Other constraints can also be represented in NIAM and are described in detail in reference [Rasdorf 90a, Nijssen 89].

3.3.1 Uniqueness Constraint

Uniqueness constraints are needed to control redundancy in a conceptual schema design. As a rule, no elementary fact may be repeated or used twice, indicating that fact instances must be unique. An entity type in a fact type can by itself be unique, meaning that no entity instance for this entity type is repeated. This results in a column with no duplicate values in the schema-based diagram. On the other hand, there are cases where no one entity type by itself is unique, but the combination of some or all of the entity types across the fact type yields unique fact instances. When an entity type is unique, it is called a *single key*, whereas when combined entity types are unique, they produce a *composite key*.

In NIAM uniqueness constraints are represented by double-sided arrows drawn on the top or bottom side of the role(s) participating in the key. Figure 5-a shows a binary fact type, **WP-Level-Of-Detail**, with the WBS-Node entity type being unique. Note the double-sided arrow on top of the *has* role played by the WBS-Node entity type. Figure 5-b shows a ternary fact type, **Material-Budgets**. The Control-Account and Material entity types form the key to this fact type. These entity types are not adjacent. Note how the arrow spans the roles played by them and is disconnected at the role played by the Quantity entity type.

The uniqueness constraint is also applied to nested fact types. One requirement for creating an objectified relationship type is that the uniqueness constraint arrow must span all the roles included in the objectified relationship and that the role played by the objectified relationship type in the nested fact type must be unique [Rasdorf 90a, Nijssen 89].

3.3.2 Mandatory and Optional Roles Constraint

Information on an input form may either be *mandatory* or *optional*. To formally specify these types of information in NIAM, the relevant roles are marked as either mandatory or optional. A role in a fact type is mandatory if every member of the population of the entity type attached to the role is required to play this role; otherwise the role is optional. A mandatory role is represented on a conceptual schema diagram by adding a bullet at the point where the arc from the role meets the entity type. If an entity type plays only one role, this role is mandatory [Nijssen 89]. Examples of the use of this constraint type can be found in Section 4.3.

3.3.3 Subtype Constraint

The subtype constraint is used when there is at least one role played by only a subset of the population of a given entity type. The role could have been marked as optional, but sometimes there is a need to emphasize the fact that only a subset of the population plays this role. To do so, a new entity type is created for the subset population. This entity type becomes a *subtype*, and the original entity type becomes a *supertype*. The subtype/supertype constraint is represented by an arrow going from the subtype entity type to the supertype one. In this setup, the subtype still plays the roles attached to the supertype, a concept called *inheritance*, in addition to the roles that are specific to the subtype. This is because the supertype still includes the members of the subtype.

Figure 6 represents this constraint in NIAM. In this figure, the roles played by the Control-Account entity type are only relevant to the population of this entity type. However, the population of the Control-Account entity type is a subset of the population of the WBS-Node (since each control account is in fact a WBS element), and the information relevant to control accounts may not be relevant to other elements on the WBS. Therefore, to overcome this problem, the Control-Account entity type is created to include this subset, and the subtype constraint is applied to it and is represented by the dark arrow going from the Control-Account entity type to the WBS-Node entity type. Note that by the subtype constraint the population of the Control-Account entity type inherits the roles played by the WBS-Node entity type.

3.4 Relational Transformation

Once the conceptual schema diagram is developed and loaded with the necessary constraints, it becomes ready to be mapped onto any computational model. The relational, hierarchical and network models are among the most popular computational data models available today. This section focuses on mapping the NIAM model onto a relational data model.

NIAM provides a relational transformation algorithm that typically guarantees the development of a fifth normal form (5NF) relational model. The algorithm is called the Optimal Normal Form (ONF) algorithm. Familiarity with the relational data model concepts is assumed in this section.

The ONF algorithm consists of three major steps [Rasdorf 90a, Nijssen 89]. These are summarized as follows:

1. A separate relation is created for each fact type that has no single key. The shortest key is selected as the primary key for the relation.
2. Fact types that share a common entity type and have single keys based on the common entity type are grouped into one relation. The primary key is selected based on this common entity type.
3. A separate relation is created for every remaining fact type, and a primary key is selected for every relation.

Moreover, when subtypes are encountered, they are mapped onto separate relations, and then the subtype relationship between the relevant attributes is expressed verbally. Additionally, objectified relationship types in nested fact types are treated as normal entity types. However, any relation created based on an objectified relationship type must have attributes related to the entity types included in that objectified relationship type. Then, once the relations are defined, the primary keys are underlined and optional attributes are marked by "OP."

4 Work Packaging and Forms Modeling Using NIAM

This section provides a construction example that uses the background information supplied in the previous section and applies the NIAM methodology to modeling the information extracted from the Means forms and the work packaging model (Sections 2.1 and 2.2). First, the elementary facts are enumerated. Then, the fact types are modeled in a step-by-step fashion to develop the complete NIAM conceptual model. The reader may want to refer back to Section 3.2.1 to refresh his memory with respect to the notation used to represent elementary facts. That notation is used in this section. This section represents the first two steps of the four-step modeling process presented in Section 1.2. The two steps are used as follows: Section 4.1 is the problem definition step and Sections 4.2 and 4.3 are the conceptual modeling step.

4.1 Elementary Facts

After the data items that are needed by the cost and schedule control functions have been identified, they are restructured to fit the work packaging model needs. The restructuring process will be presented in this section as a set of elementary facts that describes the behavior of integrated cost and schedule control within the context of the work packaging model.

Three groups of elementary facts have been developed. The first group deals with the definition of the scope of each work package and the description of OBS elements. A total of six fact types are included in this group. The following is a list of elementary facts that represents this group. The facts are extracted from Figure 4.

- **WP-Description:** WBS-Node (code) 134123 *has* Description (text) 'concrete walls in area A of floor 4.'

- **WP-Hierarchy:** WBS-Node (code) 134123 *is-child-of* WBS-Node (code) 134120.
- **WP-Level-Of-Detail:** WBS-Node (code) 134123 *has* Level-Of-Detail (number) 6.
- **OBS-Description:** OBS-Node (code) 1222 *has* Description (text) 'concrete foreman.'
- **OBS-Hierarchy:** OBS-Node (code) 1222 *is-child-of* WBS-Node (code) 1220.
- **OBS-Level-Of-Detail:** OBS-Node (code) 1222 *has* Level-Of-Detail (number) 4.

The second group of elementary facts deals with control accounts. Each control account is a node on the WBS hierarchy, and thus has an entry in group one. The WBS node code for each control account will be used as the account's code number. A total of twenty six fact types are included in this group. Group two is further broken down into three sub-groups. The first sub-group deals with the identification system, where each resource (material, manpower, and equipment) has a coding scheme that uniquely identifies its constituents in addition to the scheme that identifies task types. Also, this sub-group establishes organizational responsibilities for the control accounts and defines scheduling coding needs such as start and end events. The second sub-group deals with budgeted data for resource use. The third sub-group deals with actual data on resource use. The following is a subset of facts that represents each of these sub-groups.

1. Sub-Group 1:
    - **Material-Codes:** Material (code) 100 *has* Description (text) '3000 psi concrete.'
    - **Material-Units:** Material (code) 100 *has* Units (unit) 'cubic yard.'
    - **Craft-Codes:** Craft (code) 10 *has* Description (text) 'concrete workers.'
    - **Employee:** Worker (name) 'D. Callaway' *has* Worker-ID (code) 101.
    - **Workers'-Crafts:** Worker (name) 'D. Callaway' *has* Craft (code) 10.
    - **Equipment-Codes:** Equipment (code) 10 *has* Description (text) 'concrete pump.'
    - **Task-Codes:** Task (code) 100 *has* Description (text) 'concrete for walls.'
    - **OBS/Control-Account-Assignments:** Control-Account (code) 134123 *is-assigned-to* OBS-Node (code) 1222.
    - **Task-Codes/Control-Account-Assignments:** Control-Account (code) 134123 *has* Task (code) 100.
    - **Control-Account-Start-Event:** Control-Account (code) 134123 *has-start* Event (code) 100.
    - **Control-Account-End-Event:** Control-Account (code) 134123 *has-end* Event (code) 200.
2. Sub-Group 2:
    - **Material-Budgets:** Control-Account (code) 134123 *has-budget* Quantity (number) 100 *for* Material (code) 100.
    - **Material-Budget-Unit-Costs:** Material (code) 100 *has-budget-unit-cost* Amount (dollars) 30.
    - **Equipment-Budgets:** Control-Account (code) 134123 *has-budget* Work-Hours (hours) 20 *for* Equipment (code) 10.

- **Equipment-Budget-Hourly-Rates:** Equipment (code) 10 *has-budget-rental-rate* Amount (dollars) 20.
- **Hours-Budgets:** Control-Account (code) 134123 *has-budget* Man-Hours (man-hours) 120 *for* Craft (code) 10.
- **Daily-Man-Hours-Plan:** Control-Account (code) 134123 *has-budget* Man-Hours (man-hours) 60 *for* Craft (code) 10 *on* Day (date) 'August 1, 1990.'
- **Craft-Budget-Pay-Rates:** Craft (code) 10 *has-budget-pay-rate* Amount (dollars) 8.

3. Sub-Group 3:

- **Material-Actual-Use:** Control-Account (code) 134123 *used* Quantity (number) 30 *on* Day (date) 'August 1, 1990.'
- **Material-Actual-Unit-Costs:** Material (code) 100 *had-actual-unit-cost* Amount (dollars) 35 *on* Day (date) 'August 1, 1990.'
- **Equipment-Actual-Use:** Control-Account (code) 134123 *used* Work-Hours (hours) 8 *of* Equipment (code) 10 *on* Day (date) 'August 1, 1990.'
- **Equipment-Actual-Hourly-Rates:** Equipment (code) 10 *had-actual-rental-rate* Amount (dollars) 18 *on* Day (date) 'August 1, 1990.'
- **Regular-Actual-Hours:** Worker-ID (code) 101 *had-worked-on* Control-Account (code) 134123 *on* Day (date) 'August 1, 1990.' Activity *lasted-regular* Work-Hours (hours) 8.
- **Overtime-Actual-Hours:** Worker-ID (code) 101 *had-worked-on* Control-Account (code) 134123 *on* Day (date) 'August 1, 1990.' Activity *lasted-overtime* Work-Hours (hours) 3.
- **Workers'-Regular-Pay-Rates:** Worker (name) 'D. Callaway' *has-regular-pay-rate* Amount (dollars) 8.
- **Workers'-Overtime-Pay-Rates:** Worker (name) 'D. Callaway' *has-overtime-pay-rate* Amount (dollars) 10.

Note in the **Material-Actual-Use** fact type that the Material entity type was eliminated. This is because it is assumed that each control account tracks the consumption of one major material type for progress as well as cost control purposes. Thus, retrieving the type of material used within the scope of a given control account would rely on the **Material-Budgets** fact type.

The third group of elementary facts deals with creating the historical database from the daily accumulation of data and information about control accounts. A total of 10 fact types are included in this group. Even though it may appear that these facts duplicate the information provided by the elementary facts described above, this is not the case. This group summarizes the daily records of data and information for each control account into one historical record once the account is completed. Additionally, computed total costs are permanently recorded by the facts included in this group, whereas computed costs in the previous groups have not been considered because they are of the derived type. Such permanent historical records assist construction planners in planning and estimating future projects and are more useful than the daily records that are large in number and are not readily available for future planning and estimating of new projects. Moreover, daily records may be deleted if historical records are properly summarized and documented, and this reduces storage requirements.

Enumerated below is a subset of the elementary facts that represent this group.

- **Regular-Hours-History:** Control-Account (code) 134123 *had-workers-from* Craft (code) 10. Assignment *lasted-regular* Man-Hours (man-hours) 150.
- **Overtime-Hours-History:** Control-Account (code) 134123 *had-workers-from* Craft (code) 10. Assignment *lasted-overtime* Man-Hours (man-hours) 20.
- **Regular-Hours-Cost-History:** Control-Account (code) 134123 *had-workers-from* Craft (code) 10. Assignment *had-regular-cost* Amount (dollars) 800.
- **Overtime-Hours-Cost-History:** Control-Account (code) 134123 *had-workers-from* Craft (code) 10. Assignment *had-overtime-cost* Amount (dollars) 200.
- **Material-Use-History:** Control-Account (code) 134123 *used* Material (code) 100. Consumption *was* Quantity (number) 120.
- **Material-Cost-History:** Control-Account (code) 134123 *used* Material (code) 100. Consumption *had-cost* Amount (dollars) 4200.
- **Equipment-Use-History:** Control-Account (code) 134123 *used* Equipment (code) 10. Operation *was-for* Work-Hours (number) 25.
- **Equipment-Cost-History:** Control-Account (code) 134123 *used* Equipment (code) 10. Operation *had-cost* Amount (dollars) 450.
- **Control-Account-Actual-Start:** Control-Account (code) 134123 *started-on* Day (date) 'August 1, 1990.'
- **Control-Account-Actual-Finish:** Control-Account (code) 134123 *finished-on* Day (date) 'August 8, 1990.'

Note that the Amount entity type in both **Regular-Hours-Cost-History** and **Overtime-Hours-Cost-History** fact types is computed as the multiplication of the man-hours and the corresponding actual pay rates for each craft. Thus, different crafts within a given control account would have separate historical records, unlike the **Material-Use-History**, where one historical record is created per control account. Similarly, more than one historical record may be created per control account for tracking equipment use. From this discussion, one can observe that a control account has only one material type associated with it, but may have a number of crafts and pieces of equipment being tracked. In most cases however, a control account would also have one craft and one piece of equipment associated with it.

4.2 NIAM Conceptual Schema Diagram Development

This section provides the formal NIAM representation of the elementary facts. Each fact type is represented by an individual CSD. The fact types are addressed in the same grouping approach used in the previous section. Then, the uniqueness constraints are added to each CSD. Other constraints are delayed to the next section, where the CSD's are combined to develop the complete formal representation of the work packaging model.

Group one includes six binary fact types: **WP-Description**, **WP-Hierarchy**, **WP-Level-Of-Detail**, **OBS-Description**, **OBS-Hierarchy**, and **OBS-Level-Of-Detail**. Figure 7 shows the CSD representation of each one of these fact types, along with their uniqueness constraints. Note how each of the two homogenous binaries (**WP-Hierarchy** and **OBS-Hierarchy**) is represented as one entity type (WBS-Node or OBS-Node) that plays two roles.

Group two is broken down into three sub-groups. Sub-group one includes eleven binary fact types: **Material-Codes**, **Material-Units**, **Craft-Codes**, **Employee**, **Workers'-Crafts**, **Equipment-Codes**, **Task-Codes**, **OBS/Control-Account-Assignment**, **Task-Codes/Control-Account-Assignment**, **Control-Account-Start-Event**, and **Control-Account-End-Event**. Figure 8 shows the CSD representation of each fact type, along with their uniqueness constraints.

Sub-group two of group two includes three binary fact types, three ternary fact types, and one fact type of arity four. The binary fact types are **Material-Budget-Unit-Costs**, **Equipment-Budget-Hourly-Rates**, and **Craft-Budget-Pay-Rates**. The ternary fact types are **Material-Budgets**, **Equipment-Budgets**, and **Worker-Hours-Budgets**. The fact type with arity four is **Daily-Man-Hours-Plan**. Figure 9 shows the CSD representation of each one of these fact types, along with their uniqueness constraints.

Sub-group three of group two includes two binary fact types, three ternary fact types, one fact types of arity four, and two nested fact types. The binary fact types are **Workers'-Regular-Pay-Rates** and **Workers'-Overtime-Pay-Rates**. The ternary fact types are **Material-Actual-Use**, **Material-Actual-Unit-Costs**, and **Equipment-Actual-Hourly-Rates**. The fact type of arity four is **Equipment-Actual-Use**. The nested fact types are **Regular-Actual-Hours** and **Overtime-Actual-Hours**, where the objectified relationship is composed of three roles. Figure 10 shows the CSD representation for each one of these fact types, along with their uniqueness constraints.

Group three includes two binary facts and eight nested fact types. The binary fact types are **Control-Account-Actual-Start** and **Control-Account-Actual-Finish**. The nested fact types are **Regular-Hours-History**, **Overtime-Hours-History**, **Regular-Hours-Cost-History**, **Overtime-Hours-Cost-History**, **Material-Use-History**, **Material-Cost-History**, **Equipment-Use-History**, and **Equipment-Cost-History**. Each objectified relationship is composed of two roles. Figure 11 shows the CSD representation of each one of these fact types, along with their uniqueness constraints.

4.3 The Complete NIAM Conceptual Schema Diagram

In extracting elementary facts, care was exercised in naming entity types. Since duplicate entity types are not permitted in NIAM, similar entity types were given the same name to make the NIAM CSD representation of the complete problem easier to develop from the individual CSD's. Therefore, in combining the individual CSD's, common entity types are represented by one entity type playing a role in each of these fact types. Then, the mandatory role constraints are added to the entity types on the complete CSD. The mandatory role constraint addition was delayed to this stage because an entity type may appear to be playing a mandatory role within the context of an individual CSD, but when combined with other CSD's that share the same entity type, the mandatory role may become optional. Hence, it is more appropriate to add the mandatory role constraint to the overall CSD rather than the individual ones.

Figure 12 shows the complete NIAM CSD of the work packaging model. The figure is composed of two pages with some entity types from page one regenerated on page two for clarity purposes. The regenerated entity types are numbered 1 through 10 on both pages. When applying the mandatory constraint to any one of those ten entity types, the fact types associated with these entity types on both pages are considered at the same time. This is essential to the correct NIAM representation, since these regenerated entity types are in fact the same ones on both pages and not duplications.

In Figure 12, one subtype constraint is observed. This constraint represents the relationship between the Control-Account entity type and the WBS-Node entity type. It states that the set of instances of the Control-Account entity type is a subset of the set of instances of the WBS-Node entity type, which means that each control account is also a WBS node. This constraint isolates a portion of the WBS nodes (control accounts) from the complete set of nodes and allows for modeling the information relevant to only this portion. At the same time, the information that is pertinent to WBS nodes and modeled as fact types attached to the WBS-Node entity type is inherited by the Control-Account subtype.

Note in Figure 12 that even though the Material, Craft, Worker-ID, Equipment, Task, and Event entity types have the same reference modes, they are not combined into one entity type. The reason for this is that it is essential to emphasize that each entity type is independent from the other and represents a different coding scheme. Similarly, Level-Of-Detail and Quantity entity types have the same reference mode, "number," but are not combined into one entity type because "number" corresponds to different types of information in each entity type. In Quantity, number represents an amount, whereas in Level-Of-Detail it represents a level.

5 The Relational Schema

This section uses the ONF algorithm to transform the complete NIAM CSD (conceptual data model) to the equivalent relational database schema (computational data model). It is assumed that the reader is familiar with relational data modeling concepts.

To organize the process of relational transformation, three groups of relations are discussed. The first group, called the *static database*, includes the relations that deal with such data as budgets, definition of the WBS and OBS, and activity network definition. The second group, called the *dynamic database*, includes the relations that deal with the daily data and information acquired by field personnel on resource and time consumptions. The third group, called the *historical database*, includes the relations that deal with the creation of historical data records for future planning.

5.1 The Static Database

Referring back to Figure 12, one can see a number of fact types that fall under the static database category. Each fact type will be visited in the order imposed by the three-step ONF algorithm. First, consider step one of the ONF algorithm. Four fact types satisfy this step: **Material-Budgets**, **Equipment-Budgets**, **Hours-Budgets**, and **Daily-Man-Hours-Plan**. Each fact type is transformed into a separate relation. Each relation is given a name representing the relevant fact type name with attribute names chosen from the equivalent entity type names and their roles. The relations are the following:

- Control-Account-Budget-Materials(Control-Account-Code, Material-Code, Quantity)
- Control-Account-Budget-Hours(Control-Account-Code, Craft-Code, Man-Hours)
- Control-Account-Budget-Equipment(Control-Account-Code, Equipment-Code, Work-Hours)

- Control-Account-Daily-Man-Hours-Plan(Control-Account-Code, Date, Man-Hours)

Next, consider step two of the ONF algorithm. The WBS-Node entity type is common to and plays a unique role in the **WP-Level-Of-Detail**, **WP-Hierarchy**, and **WP-Description** binary fact types. Thus, the three fact types are grouped into the following relation:

- Work-Package-Catalog(WBS-Code, Parent-Code OP, Level-Of-Detail, Description)

Note that the primary key for this relation is the WBS-Code attribute which represents the common entity type. Also note that the Parent-Code attribute is optional, since not every WBS node has a parent node (e.g. the root). This is obvious from the optional roles played by WBS-Node entity types in the homogenous **WP-Hierarchy** fact type.

Continuing with step two, observe that the OBS-Node entity type is common to and plays a unique role in the **OBS-Level-Of-Detail**, **OBS-Hierarchy**, and **OBS-Description** binary fact types. Thus, the three fact types are grouped into the following relation:

- OBS-Catalog(OBS-Code, Parent-Code OP, Level-Of-Detail, Description)

Note that the primary key for this relation is the OBS-Code attribute which represents the common entity type. Also note that the Parent-Code attribute is optional, since not every OBS node has a parent node (e.g. the root). This is obvious from the optional roles played by OBS-Node entity types in the homogenous **OBS-Hierarchy** fact type.

Next, and still in step two, the Control-Account entity type is common to and plays a unique role in **OBS/Control-Account-Assignments**, **Task-Codes/Control-Account-Assignments**, **Control-Account-Start-Event**, **Control-Account-End-Event**, **Control-Account-Actual-Start**, and **Control-Account-Actual-Finish**. Because this set includes static fact types as well as dynamic ones (the fact types that have 'actual' in their name), it can be listed under either the static database or the dynamic database. However, since there are more static fact types than there are dynamic ones, the relation that groups them is listed under the static database. The relation is

- Control-Account-General-Information(Control-Account-Code, OBS-Code, Task-Code, Start-Event, End-Event, Actual-Start-Date, Actual-Finish-Date)

The final sets of fact types that satisfies step two are as follows: **Material-Codes**, **Material-Units**, and **Material-Budget-Unit-Costs** (with Material as the common entity type); **Craft-Codes** and **Craft-Budget-Pay-Rates** (with Craft as the common entity type); **Equipment-Codes** and **Equipment-Budget-Hourly-Rates** (with Equipment as the common entity type); and **Employee**, **Workers'-Crafts**, **Workers'-Regular-Pay-Rate**, **Workers'-Overtime-Pay-Rate** (with Worker as the common entity type). Each set of the fact types listed above is grouped into one relation based on the common entity type. The following are the resulting relations:

- Material-Codes(Material-Code, Description, Units, Budget-Unit-Cost)
- Equipment-Codes(Equipment-Code, Description, Budget-Hourly-Rate)
- Craft-Codes(Craft-Code, Description, Budget-Pay-Rate)

- Worker-Information(Worker-ID, Worker-Name, Craft-Code, Regular-Pay-Rate, Overtime-Pay-Rate)

Finally, consider step three of the ONF algorithm. There is only one fact that remains for this step, and this is **Task-Codes**. This is transformed into the following relation:

- Task-Codes(Task-Code, Description)

Figure 13 shows the use of three of the above listed relations, namely, WP-Catalog, Control-Account-Material-Budgets, and Material-Codes. The data shown in the tables are extracted from Figure 4 and from the elementary facts of group two described earlier.

5.2 The Dynamic Database

Referring back to Figure 12, note that a number of fact types fall under the dynamic database category. Each fact type will be visited in the order imposed by the three-step ONF algorithm. First, consider step one of the ONF algorithm. Four fact types satisfy this step: **Material-Actual-Use**, **Equipment-Actual-Use**, **Material-Actual-Unit-Costs**, and **Equipment-Actual-Hourly-Rates**. Each fact type is transformed into a separate relation. Each relation is given a name representing the relevant fact type name with attribute names chosen from the equivalent entity type names and their roles. The relations are the following:

- Control-Account-Actual-Materials(Control-Account-Code, Date, Quantity)
- Control-Account-Actual-Equipment(Control-Account-Code, Equipment-Code, Date, Work-Hours)
- Material-Actual-Unit-Costs(Material-Code, Date, Unit-Cost)
- Equipment-Actual-Hourly-Rates(Equipment-Code, Date, Hourly-Rate)

Next, consider step two. The two nested fact types, **Regular-Actual-Hours** and **Overtime-Actual-Hours**, satisfy this step. They share the Activity objectified relationship type. Since the objectified relationship type is treated as a normal entity type and since it plays a unique role in both nested fact types, the two nested fact types are grouped into one relation that contains attributes representing all entity types involved, including those playing roles in the objectified relationship type. The resulting relation is

- Control-Account-Actual-Hours(Control-Account-Code, Worker-ID, Date, Regular-Hours OP, Overtime-Hours OP)

Note that since both Regular-Hours and Overtime-Hours are optional attributes (marked by 'OP'), it is not necessary to record data for both attributes at the same time. This information was modeled on the CSD using the disjunction mandatory role, which is played by the Activity objectified relationship type in the two nested fact types. Also note that the key to this relation is composed of the roles comprising Activity.

Finally, no fact types satisfying step three are identified as belonging to the dynamic database.

5.3 The Historical Database

Referring back to Figure 12, note that a number of fact types fall under the historical database category. Each fact type will be visited in the order imposed by the three-step ONF algorithm. In considering step one of the ONF algorithm, no fact types were identified. Next, consider step two. The four nested fact types, **Regular-Hours-History**, **Overtime-Hours-History**, **Regular-Hours-Cost-History**, and **Overtime-Hours-Cost-History** satisfy this step. They share the Assignment objectified relationship type that plays a unique role in every one of them. Thus, the four nested fact types are grouped into one relation that contains attributes representing all entity types involved, including those playing roles in the objectified relationship type. The resulting relation is

- Control-Account-Historical-Hours(Control-Account-Code, Craft-Code, Regular-Man-Hours, Overtime-Man-Hours, Regular-Cost, Overtime-Cost)

Note that the key to this relation is composed of the roles comprising the Assignment objectified relationship type.

Similarly, **Material-Use-History** and **Material-Cost-History** nested fact types are grouped into one relation. Also, the **Equipment-Use-History** and **Equipment-Cost-History** nested fact types are grouped into one relation. The two resulting relations are

- Control-Account-Historical-Materials(Control-Account-Code, Material-Code, Quantity, Cost)
- Control-Account-Historical-Equipment(Control-Account-Code, Material-Code, Work-Hours, Cost)

Note that the keys to these two relations are composed of the roles comprising Consumption and Operation objectified relationship types, respectively.

6 Conclusions

The automated information management system described in this paper represents a significant contribution in integrating cost and schedule control functions and in solving the problems relating to the quality, integrity, and timeliness of data. In seeking an automated solution, we utilized the work packaging model, a sound, existing integrated cost and schedule control model, and formally and systematically designed automated mechanisms to support it. Such a formal methodological design approach is lacking in most of the engineering database development efforts, though it is essential to the success of the information management system design. We believe it can have a substantial impact in the construction industry.

To achieve automation, we identified and used a formal methodology (NIAM) to develop a conceptual data model. NIAM was used to represent and subsequently automate the work packaging model and the data items extracted from the Means forms. The methodology is a straight forward, systematic approach to data modeling and database design, and is a natural approach to semantic modeling. NIAM methodology adopts the data-oriented approach to modeling for database design, which relies on inputs to the system. The Means forms were used for analyzing inputs to construction cost and schedule control systems in addition to the requirements of the work packaging model.

NIAM models are independent of any computational data models and can be mapped onto any one of them. The mapping to the relational data model was emphasized in this paper. An algorithm for mapping to the relational data model, known by the ONF algorithm, was used in the paper. The ONF algorithm produces 5NF relational models. It is believed that the relational computer data model provides the necessary, standardized, and widely accepted automated mechanisms for data storage and retrieval.

By combining the automation of information management with the automation of data acquisition in the work packaging model, we hope not only to solve the problems outlined in the paper's introduction, but also to lay a foundation for integrating other managerial functions, such as material management, with cost and schedule control. However, further investigation is needed regarding the coupling of automated information management and data acquisition systems.

Acknowledgements

This work was sponsored by the National Science Foundation under Grant MSM-8451465, a Presidential Young Investigator Award. The support of this organization is gratefully acknowledged. The authors would also like to express their gratitude to Dr. C. William Ibbs of the University of California at Berkeley, Dr. Paul Teicholz of the Center for Integrated Facility Engineering at Stanford University, Dr. Chris Hendrickson of Carnegie Mellon University, Dr. Leland Riggs of the Georgia Institute of Technology, Dr. Leonard Bernold of the University of Maryland, and Dr. Francois Grobler for their comments and suggestions on the research effort addressed in this paper and for some of the reference materials cited in the paper.

References

[Abudayyeh 90] Abudayyeh, O.Y., and Rasdorf, W.J.,"An Assessment of Cost and Schedule Control Integration Models," Technical Report Number CE-90-10, Civil Engineering Department, North Carolina State University, August, 1990.

[Adrian 86] Adrian, J.J., *Construction Accounting*," A Reston Book, Prentice-Hall, Englewood Cliffs, NJ, 1986.

[Badger 87] Badger, A.A., Reinschmidt, K.A., and Gandt, A.R., "Project Control System Integration," *Project Controls: Needs and Solutions, Proceedings of the Speciality Conference,* Sponsored by the Construction Division of the American Society of Civil Engineers, Edited by Ibbs, C.W. and Ashley, D.B., Chicago, IL, Pages 88-100, June, 1987.

[Barrie 84] Barrie, D.S. and Paulson, B.C., *Professional Construction Management*, McGraw-Hill Series in Construction Engineering and Project Management, McGraw-Hill Book Company, New York, NY, 1984.

[Carlsen 78] Carlsen, R.D. and McHugh, J.F., *Handbook of Construction Operations Forms and Formats*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1978.

[CII 88]	"The Work Packaging for Project Control," The Construction Industry Institute, Publication Number 6-6, November, 1988.
[CII 87]	"Project Control for Construction," The Construction Industry Institute, Bureau of Engineering Research, the University of Texas at Austin, Publication 6-5, September, 1987.
[Date 90]	Date, C.J., *An Introduction to Database Systems*, Volume 1, Fifth Edition, The Systems Programming Series, Addison-Wesley, Reading, MA, 1990.
[DOE 80]	The U.S. Department of Energy, "Cost and Schedule Control Systems Criteria for Contract Performance Measurement: Implementation Guide," Document Number DOE/CR-0015, May, 1980.
[Eaton 90]	Eaton, D., "Enhancing a Relational Database: Stepwise, Bottom-up Design," *Proceedings of the 1990 North American INGRESS Users Association Conference,* Salt Lake City, Utah, May, 1990.
[Finkelstein 89]	Finkelstein, R., "Database Design with NIAM," *Database Programming and Design Magazine*, Pages 15-16, February, 1989.
[Halpin 87]	Halpin, D.W., Escalona, A.L., and Szmurlo, P.M., "Work Packaging for Project Control," Source Document Number 28, Construction Industry Institute, Austin, TX, August, 1987.
[Halpin 85]	Halpin, D.W., *Financial and Cost Concepts for Construction Management*, John Wiley and Sons, New York, NY, 1985.
[Hendrickson 89]	Hendrickson, C.T. and Au, T., *Project Management for Construction: Fundamental Concepts for Owners, Engineers, Architects, and Builders,* Prentice Hall, Englewood Cliffs, NJ, 1989.
[Ibbs 87]	Ibbs, C.W., Ashley, D.B., Neil, J.M., and Feiler, F.W., "An Implementation Strategy for Improving Project Control Systems," *Project Controls: Needs and Solutions, Proceedings of the Speciality Conference,* Sponsored by the Construction Division of the American Society of Civil Engineers, Edited by Ibbs, C.W. and Ashley, D.B., Chicago, IL, Pages 101-112, June, 1987.
[Kratt 89]	Kratt, T.J., "Poor Communication of Schedule and Cost Control Data Between Construction Site Management and Field Supervision Improved by the Weekly Cost and Schedule Meeting for Direct Labor Supervisors," *Proceedings of the First Construction Congress,* American Society of Civil Engineers, San Francisco, CA, Pages 387-390, March, 1989.
[Means 86]	*Means Forms for Building Construction Professionals*, R.S. Means Company, Kingston, MA, 1986.
[Neil 83]	Neil, J.M., "A System for Integrated Project Management," *Proceedings of the Conference on Current Practice in Cost Estimating and Cost Control,* American Society of Civil Engineers, Austin, TX, Pages 138-146, April, 1983.
[Nijssen 89]	Nijssen, G.M. and Halpin, T.A., *Conceptual Schema and Relational Database Design: A Fact-Oriented Approach*, Prentice Hall, New York, NY, 1989.

[Oracle 88] "Conceptual Design and Building of Relational Databases," *Oracle Magazine*, Volume 2, Number 2, Pages 65-69, Fall, 1988.

[Rasdorf 90a] Rasdorf, W.J. and Abudayyeh, O.Y.,"An Introduction to NIAM Conceptual Schema Modeling for Database Design Using Construction Management Examples," Submitted to the *Journal of Engineering with Computers*, September, 1990.

[Rasdorf 90b] Rasdorf, W.J., and Abudayyeh, O.Y., W.J.,"An Assessment of Cost and Schedule Control Integration Models," Submitted to the *Journal of Construction Engineering and Management*, July, 1990.

[Rasdorf 87] Rasdorf, W.J., "Extending Database Management Systems for Engineering Applications," *Journal of Computers in Mechanical Engineering*, American Society of Mechanical Engineers, Volume 5, Number 5, Pages 62-69, March, 1987.

[Riggs 87] Riggs, L.S., "Project Control Techniques," *Project Controls: Needs and Solutions, Proceedings of the Speciality Conference,* Sponsored by the Construction Division of the American Society of Civil Engineers, Edited by Ibbs, C.W. and Ashley, D.B., Chicago, IL, Pages 11-25, June, 1987.

[Sharp 90] Sharp, J.K, "Information Engineering: Sandia's Computer Integrated Manufacturing (CIM) Database," *Proceedings of the ASME International Computers in Engineering Conference and Exposition*, American Society of Mechanical Engineers, Boston, MA, Pages 93-99, August, 1990.

[Sharp 89] Sharp, J.K, Orman, J.L., and Stevens, N.H., "Information Engineering: How Sandia is Developing the CIM Databases (the NIAM Approach)," Technical Report Number SAND89-0532C, Sandia National Laboratories, Albuquerque, NM, 1989.

[Teicholz 87] Teicholz, P.M., "Current Needs for Cost Control Systems," *Project Controls: Needs and Solutions, Proceedings of the Speciality Conference*, Sponsored by the Construction Division of the American Society of Civil Engineers, Edited by Ibbs, C.W. and Ashley, D.B., Chicago, IL, Pages 47-57, June, 1987.

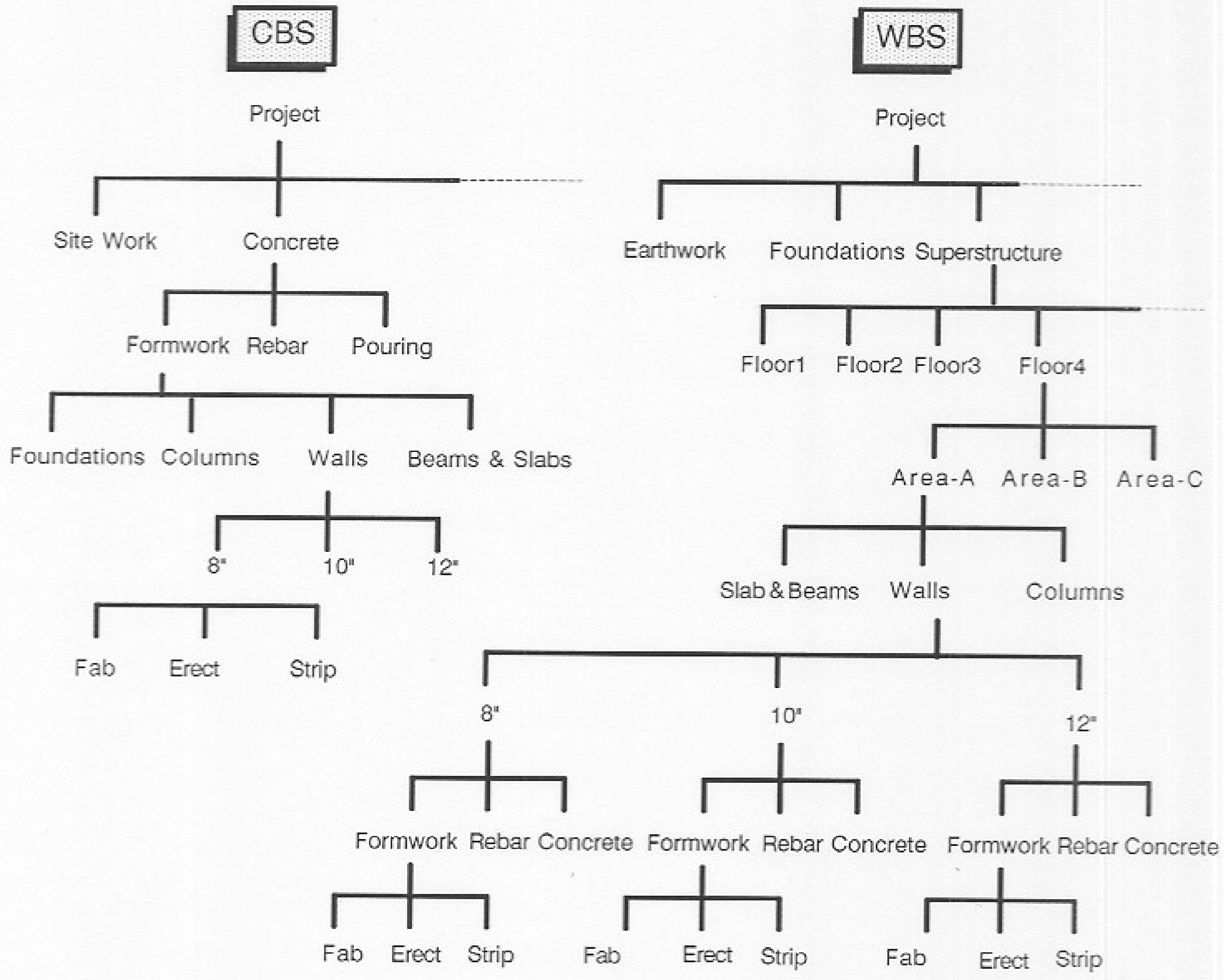


Figure 1: The Difference Between CBS and WBS

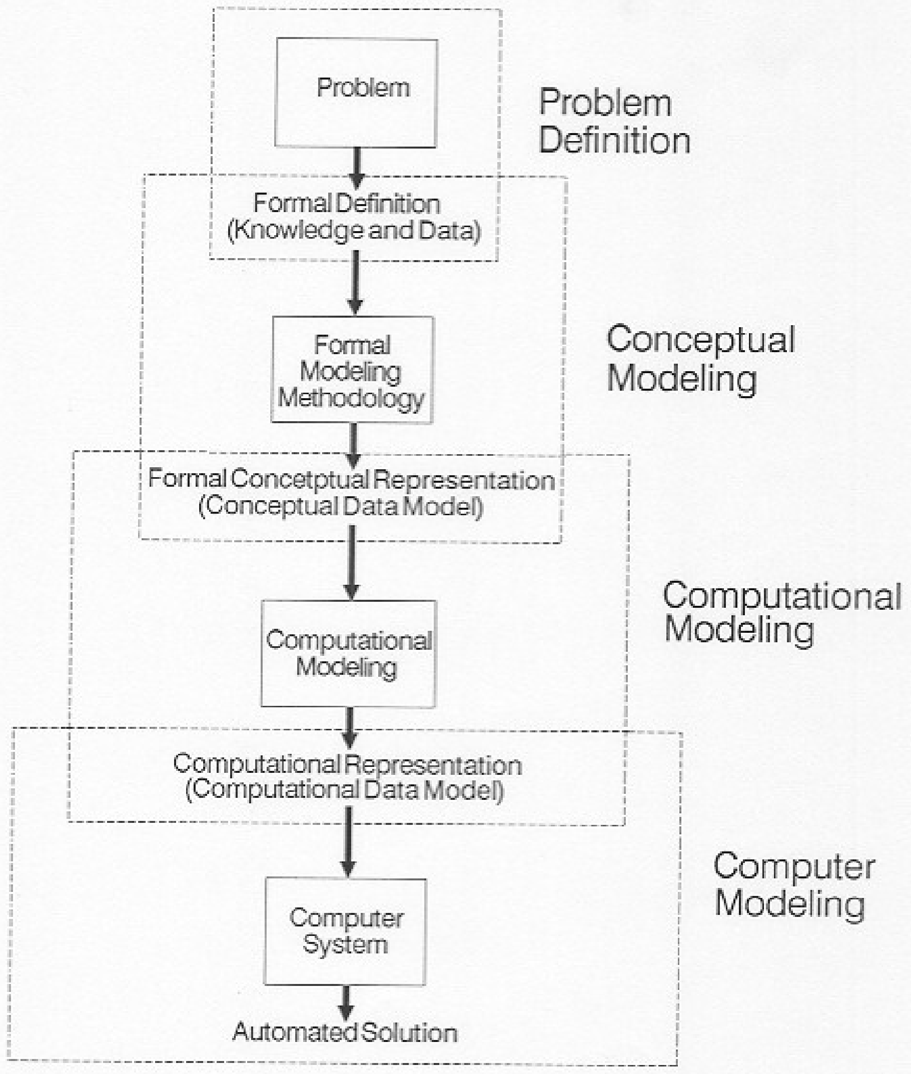

Problem
Problem Definition
Formal Definition (Knowledge and Data)
Formal Modeling Methodology
Conceptual Modeling
Formal Concetptual Representation (Conceptual Data Model)
Computational Modeling
Computational Modeling
Computational Representation (Computational Data Model)
Computer System
Computer Modeling
Automated Solution

Figure 2: The Engineering Problem Modeling Process

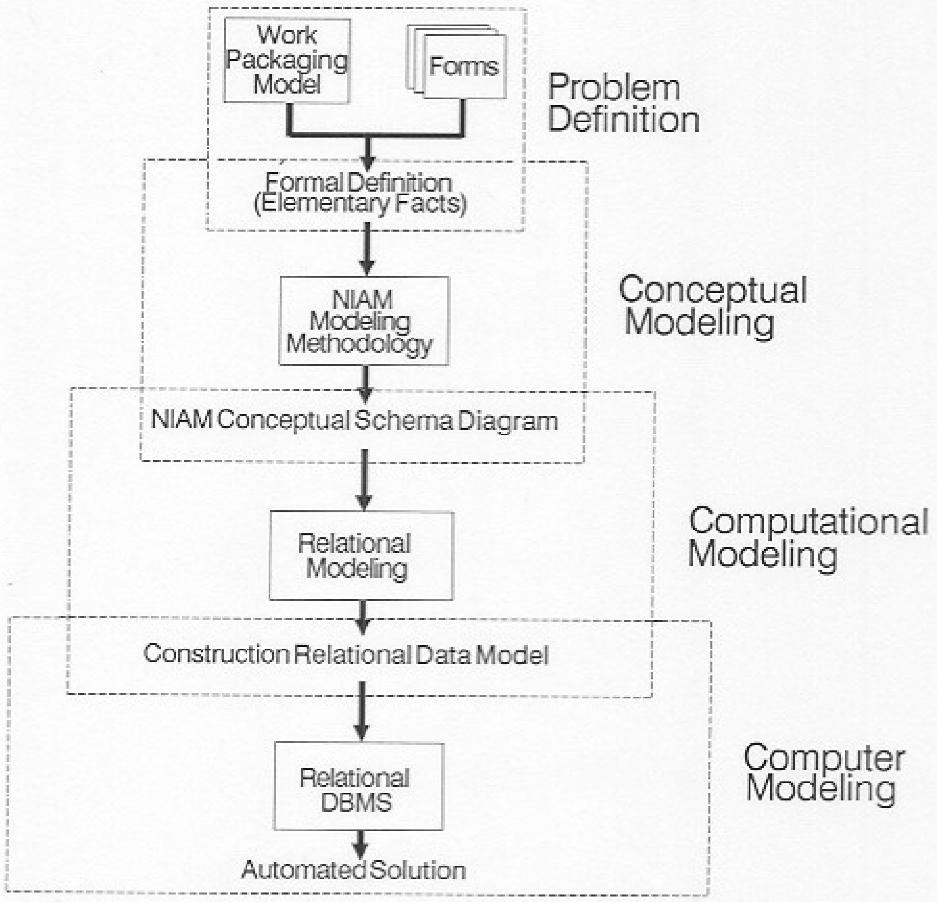


Figure 3: A Construction Problem Modeling Example

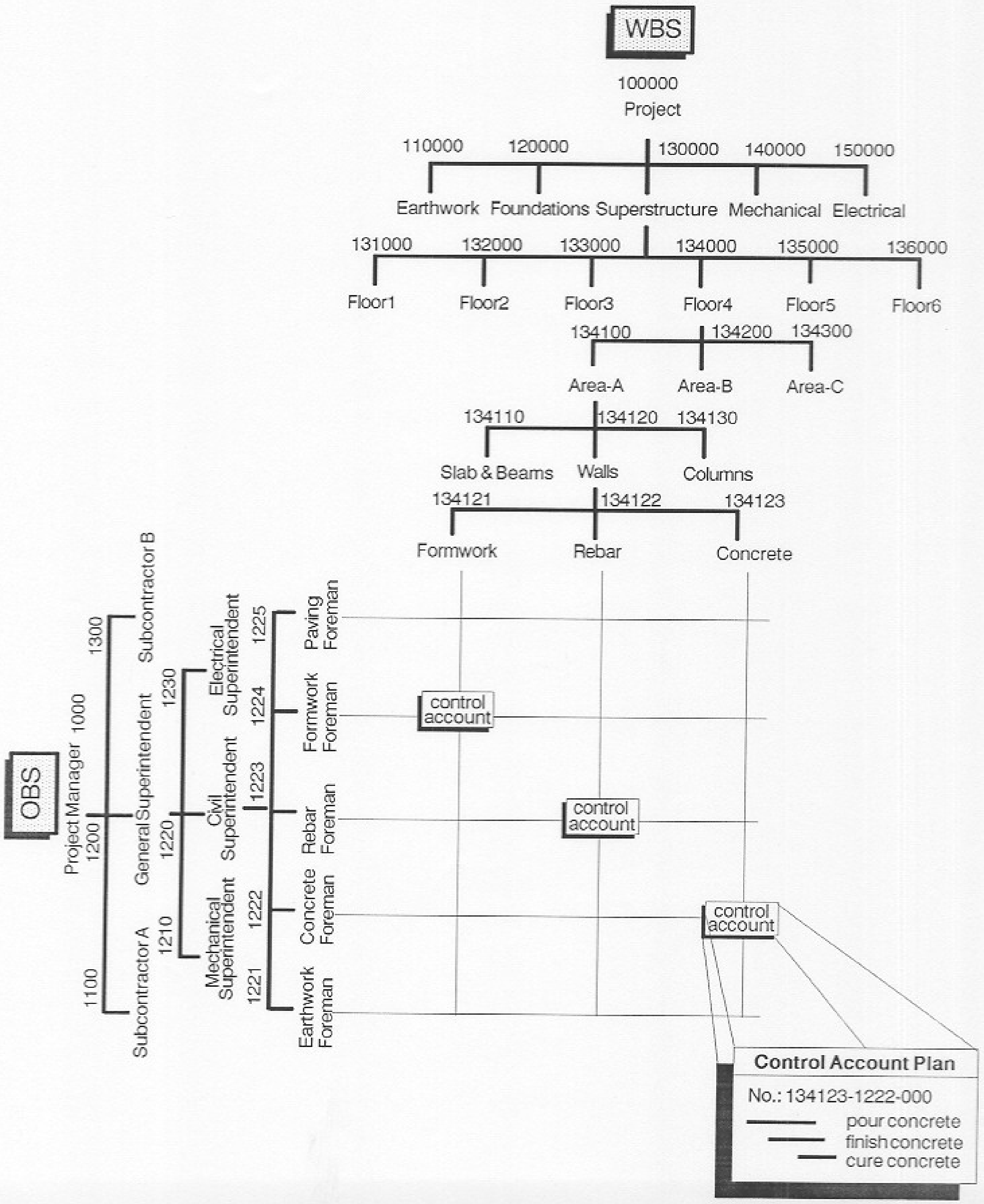
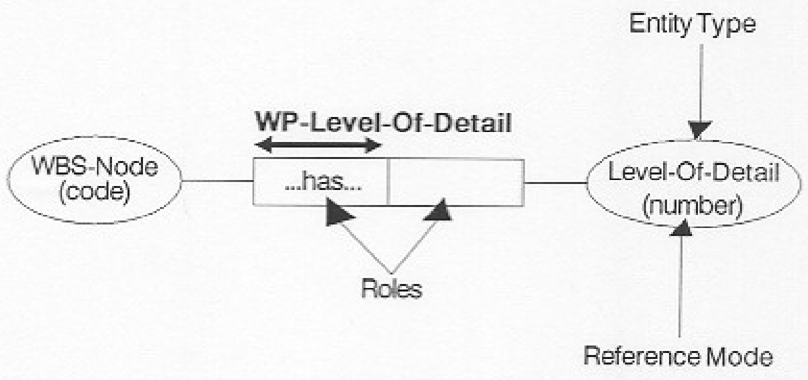
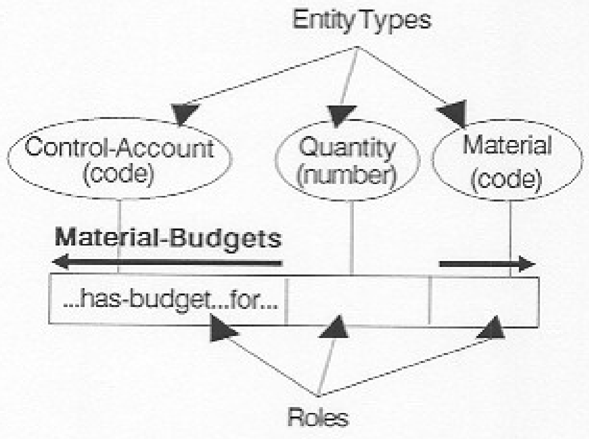


Figure 4: The Integration of the WBS and OBS



(a) Binary



(b) Ternary

Figure 5: Examples of Fact Types:

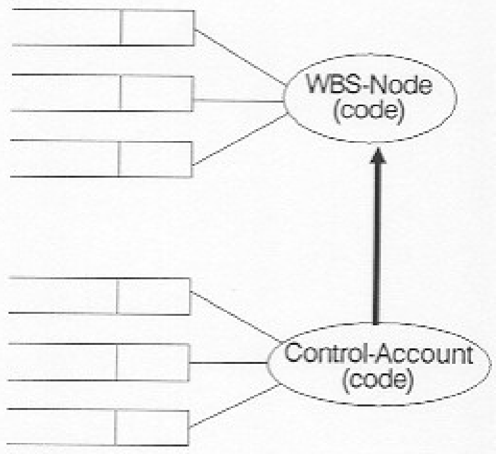


Figure 6: The Subtype Constraint

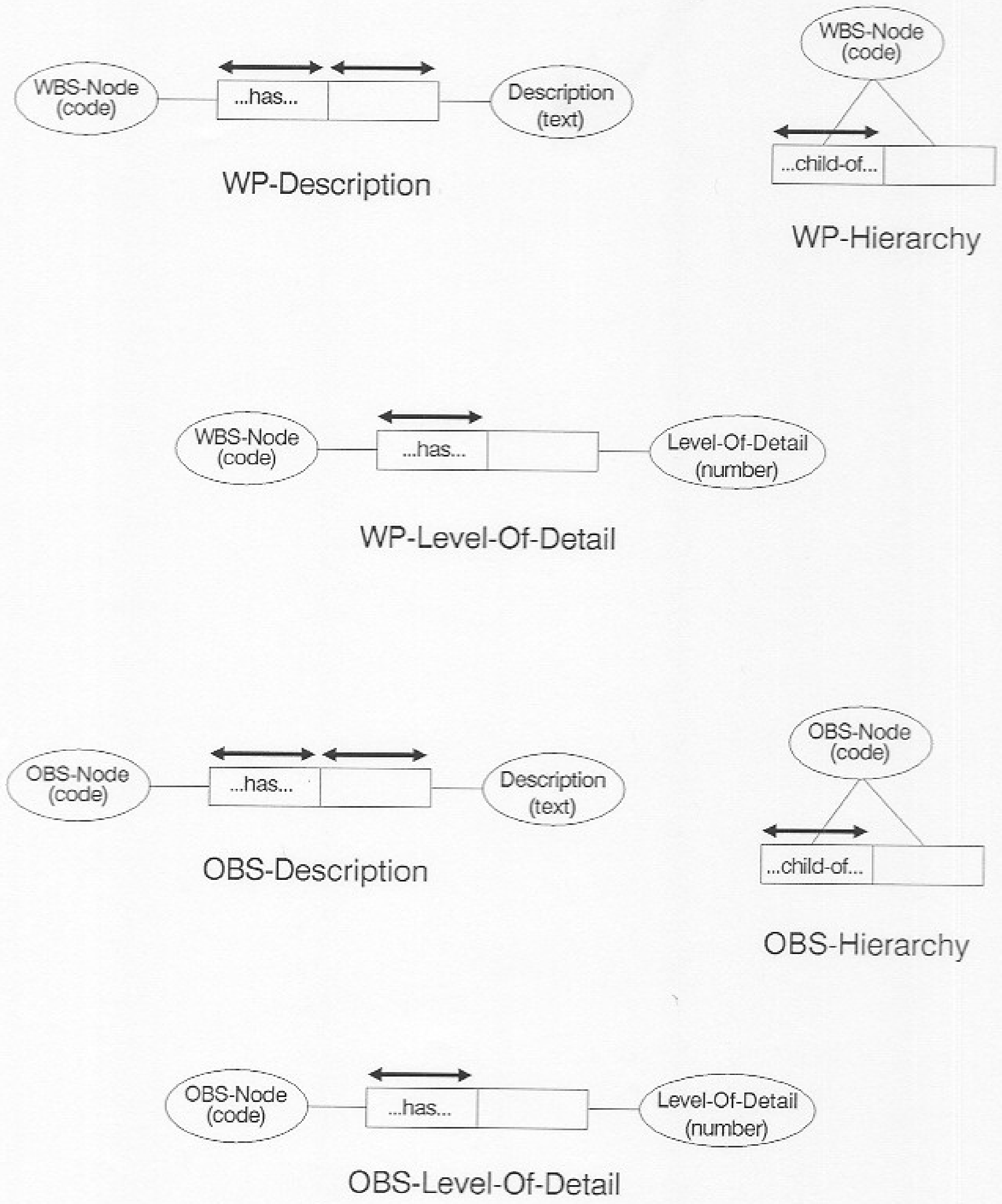


Figure 7: Group One Facts

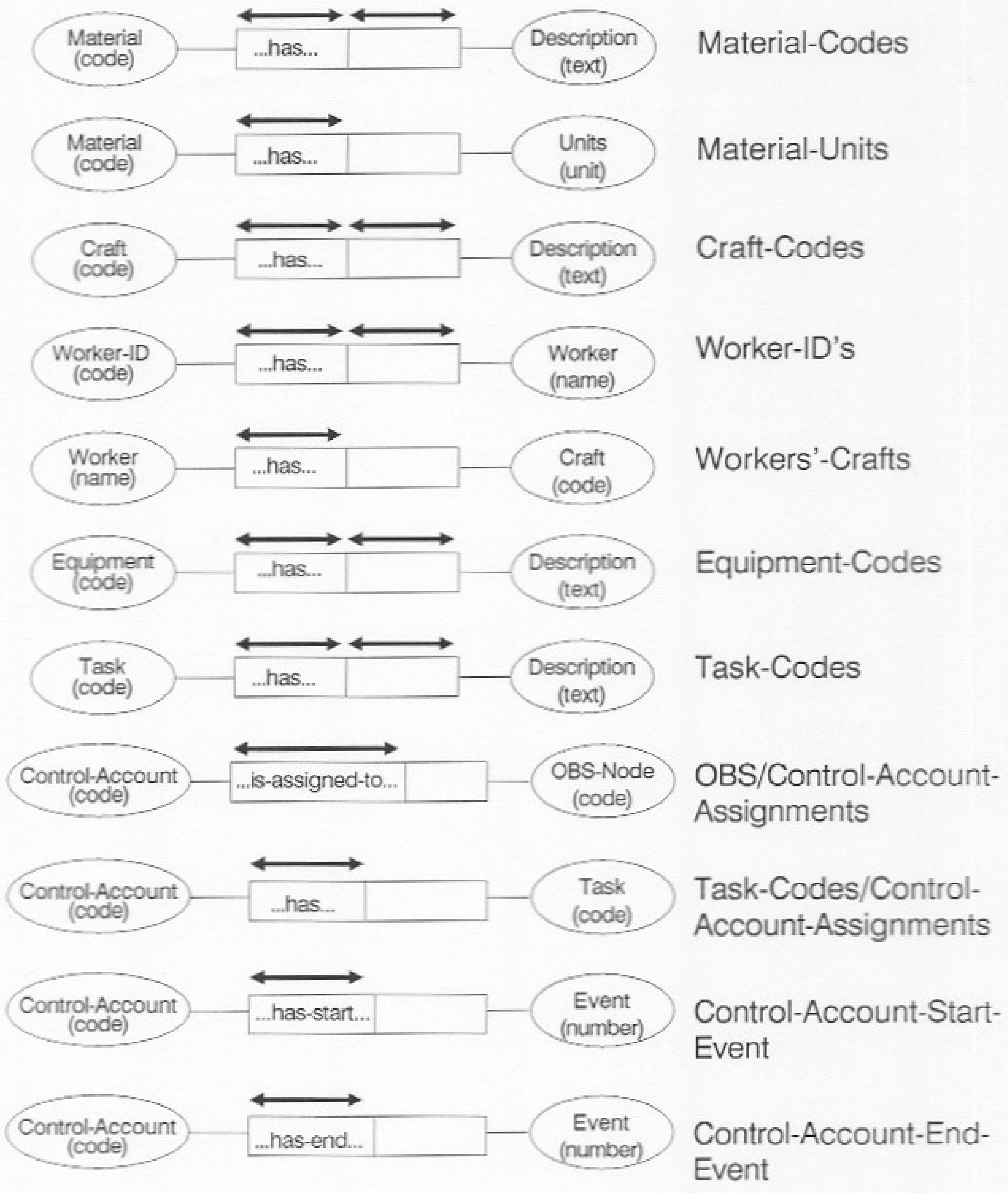


Figure 8: Group Two Facts - Sub-Group 1

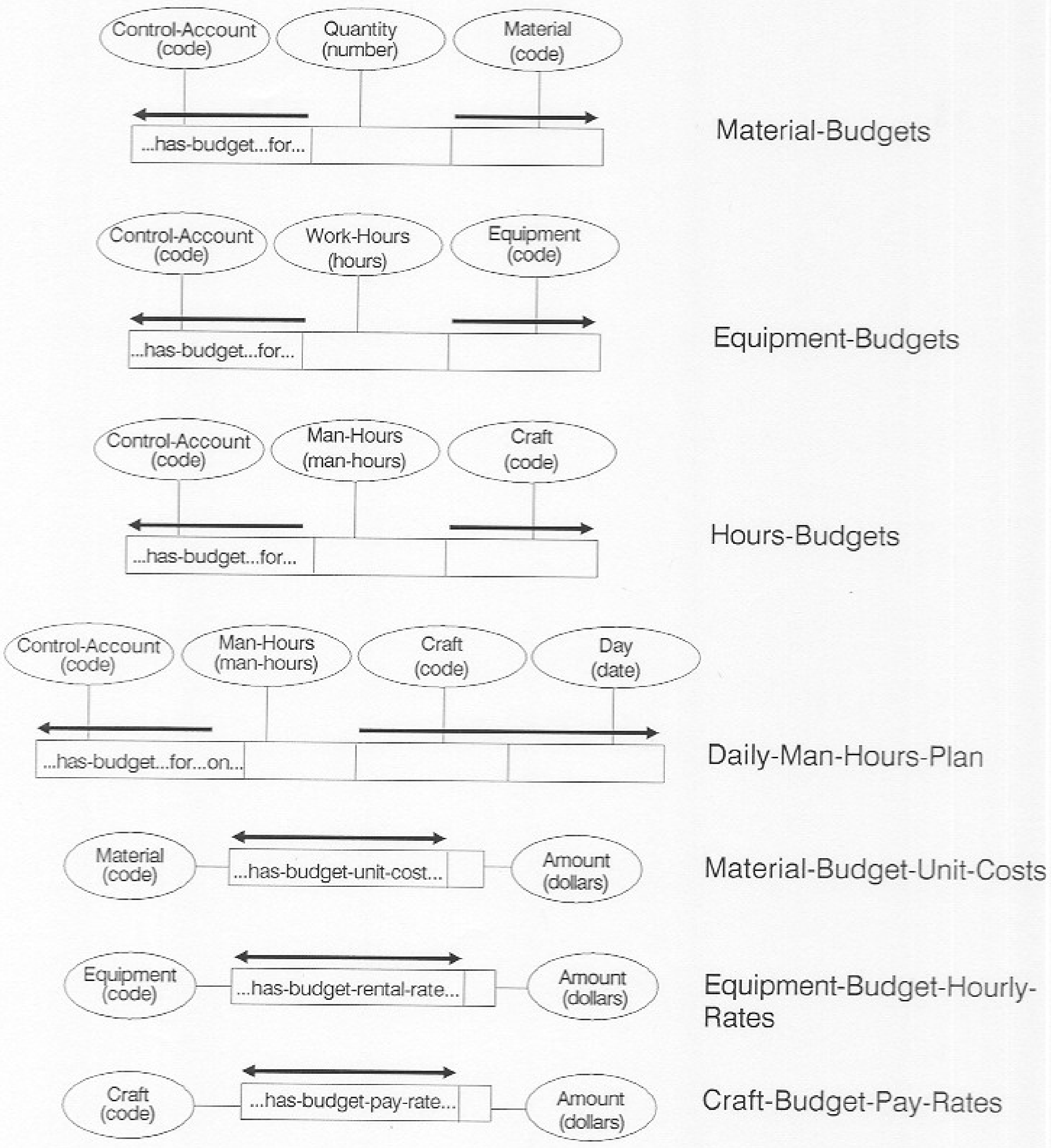


Figure 9: Group Two Facts - Sub-Group 2

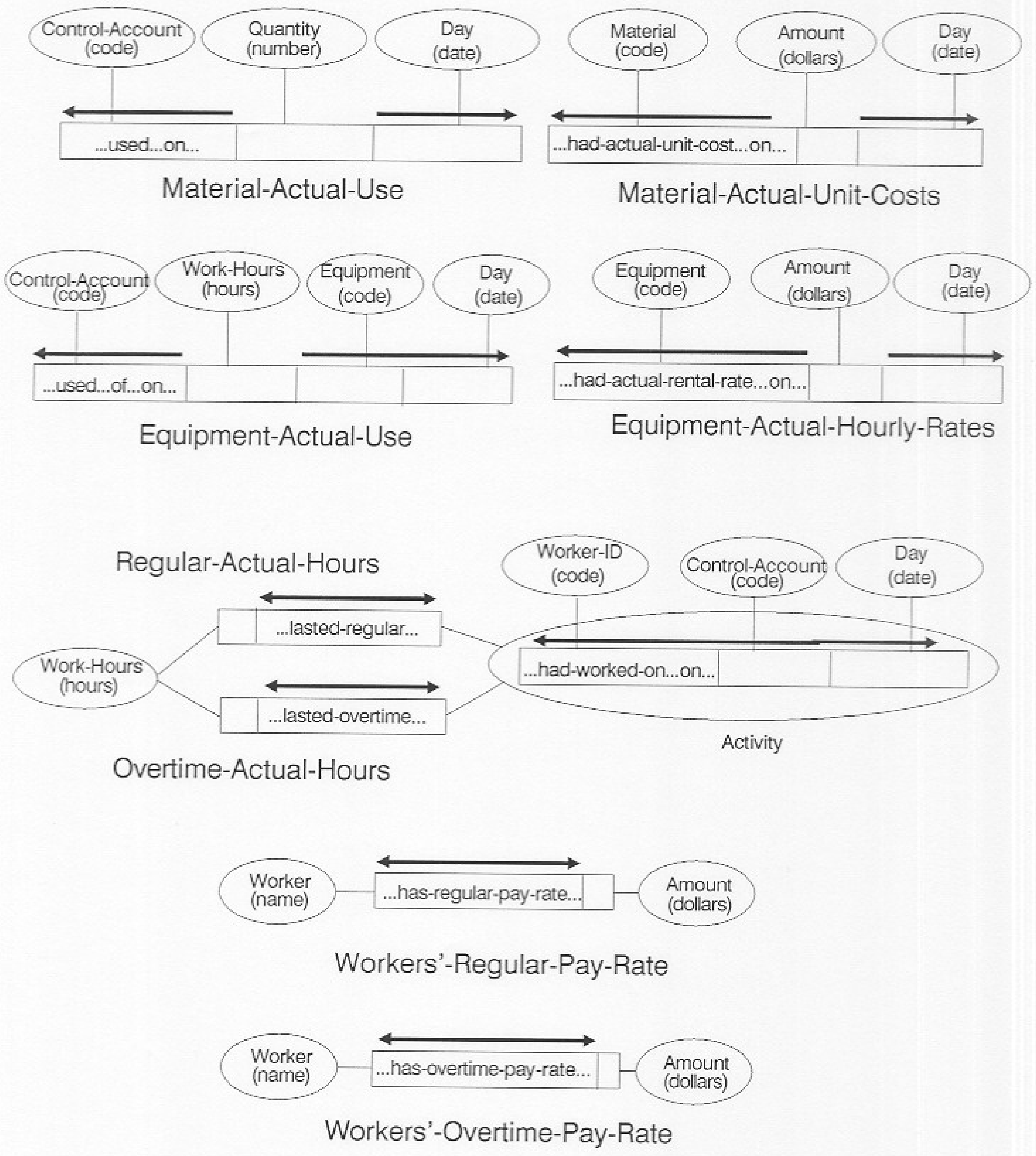


Figure 10: Group Two Facts - Sub-Group 3

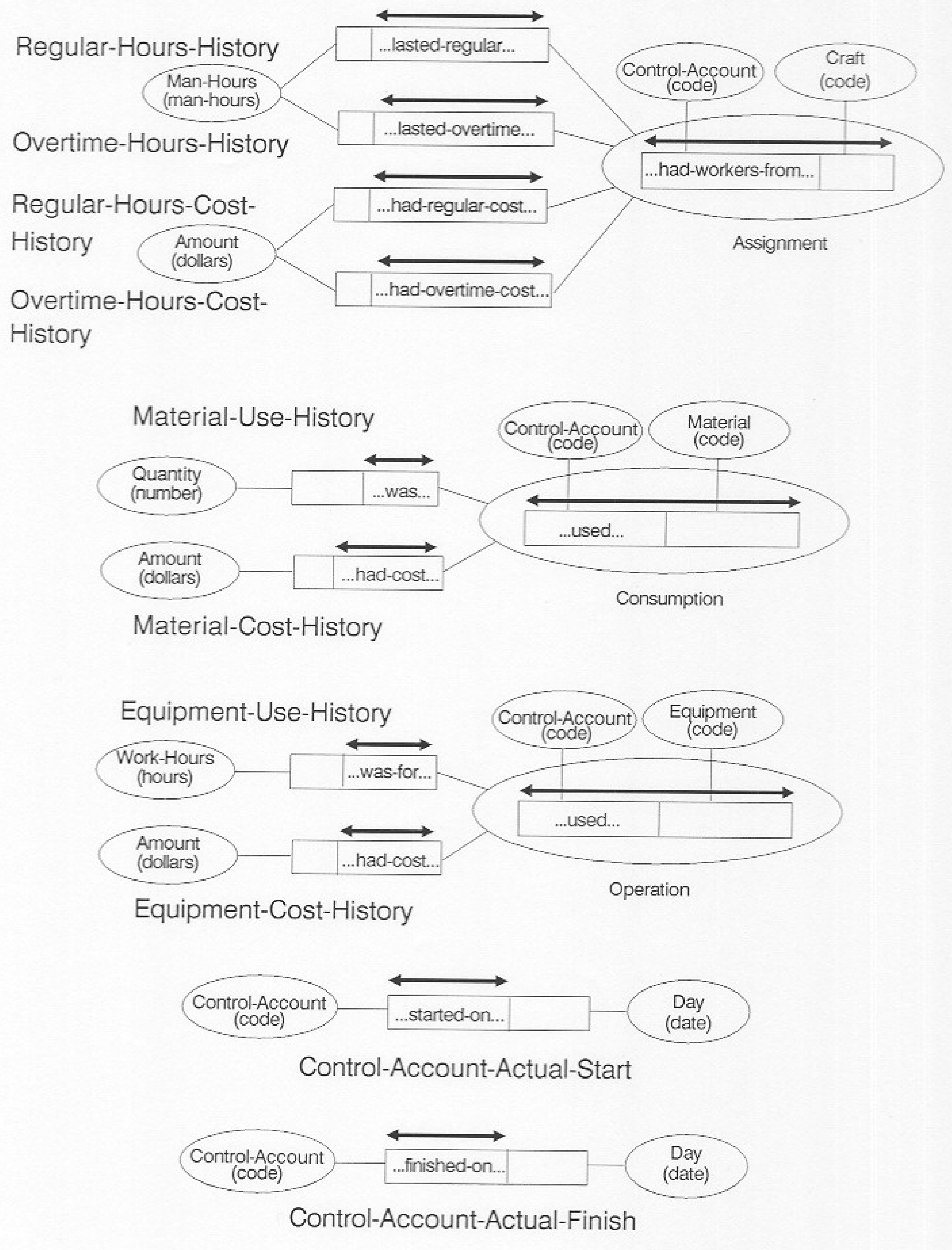


Figure 11: Group Three Facts

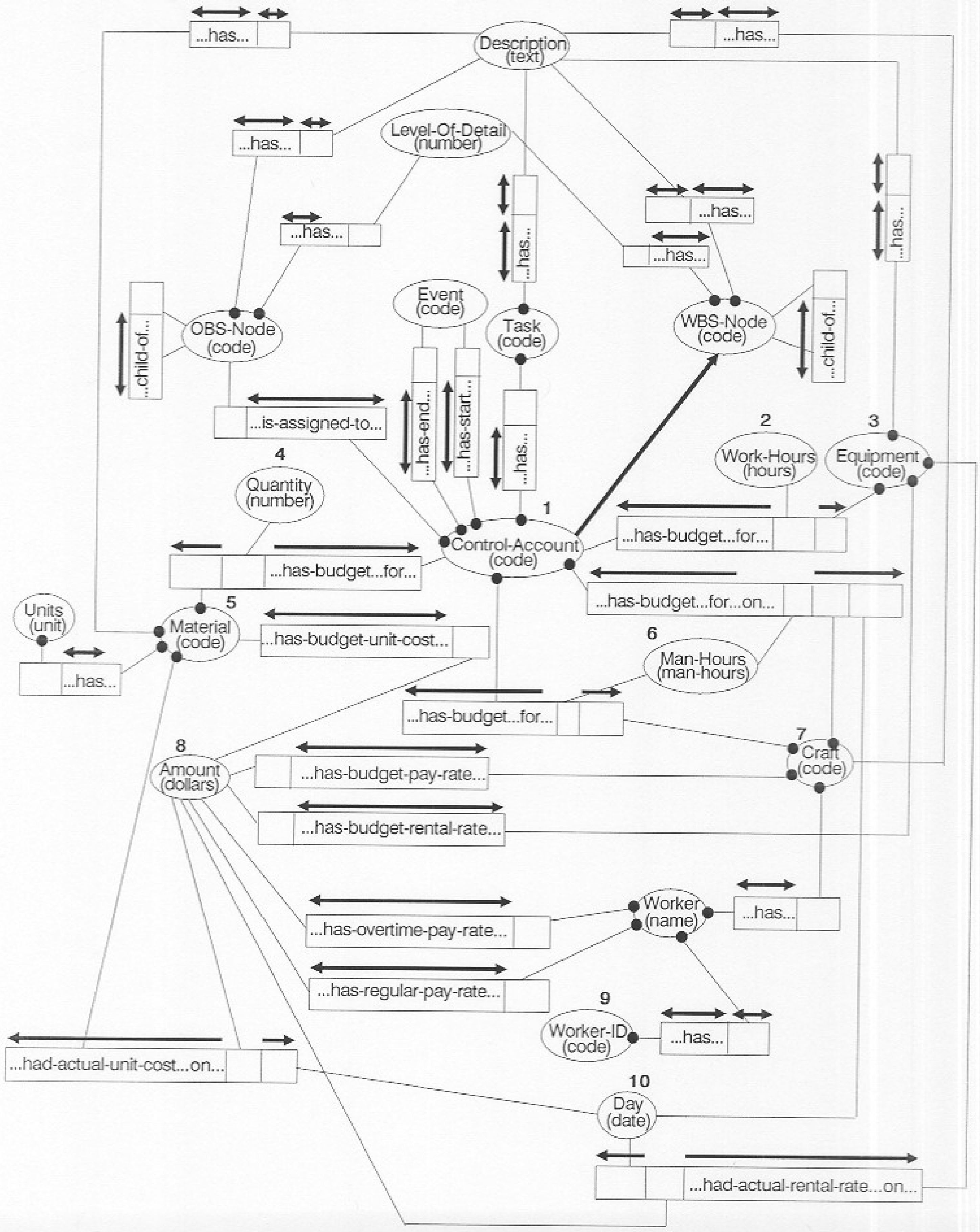


Figure 12: The Complete CSD (page 1/2)

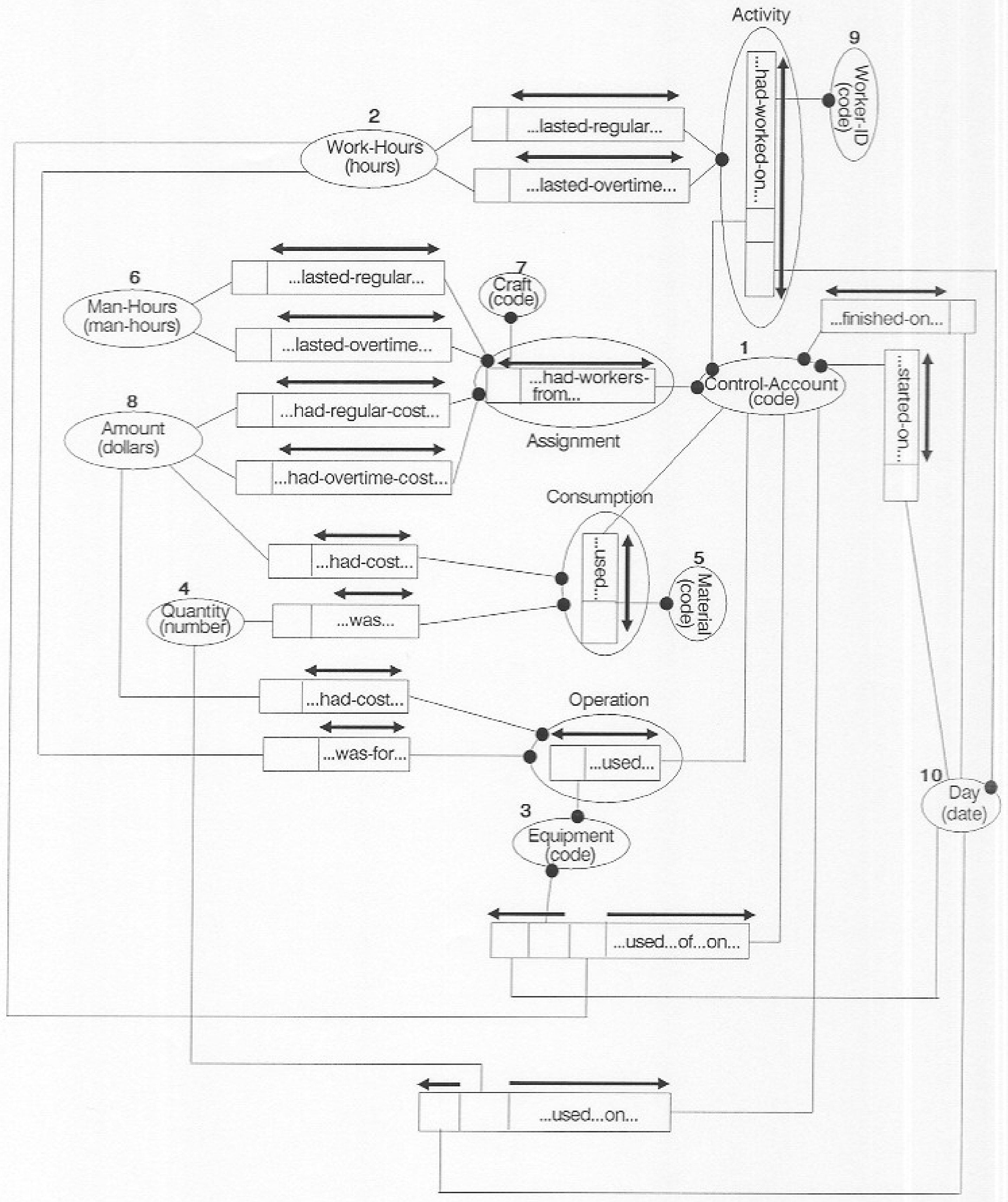


Figure 12 (continued): The Complete CSD (page 2/2)

WP-Catalog

WBS-Node	Parent-Code	Level-Of-Detail	Description
134120	134120	5	walls in area A on floor 4
134121	134120	6	formwork for walls in area A on floor 4
134122	134120	6	rebar for walls in area A on floor 4
134123	134120	6	concrete for walls in area A on floor 4

Control-Account-Budget-Materials

Control-Account-Code	Material-Code	Quantity
134121	100	400
134122	110	1000
134123	120	10000

Material-Codes

Material-Code	Description	Units	Budget-Unit-Cost
100	timber forms	sf	1.00
110	60 ksi rebars	lb	0.40
120	3000 psi concrete	cy	30.00

Figure 13: Examples of Three Relations