A resilient and scalable service architecture for the Smart Grid

Filipe Campos, Miguel Matos, Jose Pereira, Rui Oliveira´
High-Assurance Software Laboratory,
INESC TEC & University of Minho,
Braga, Portugal
Email: {fcampos, miguelmatos, jop, rco}@di.uminho.pt

Abstract—The Smart Grid vision needs to address hard challenges such as interoperability, reliability and scalability before it can become fulfilled. The need to provide full interoperability between current and future energy and non-energy systems and its disparate technologies along with the problem of seamless discovery, configuration, and communication of a large variety of networked devices ranging from the resource constrained sensing devices to the large machines inside a data center calls for an agnostic Service Oriented Architecture. Moreover, the sheer scale of the Smart Grid and the criticality of the communication among its subsystems for proper management, demands a scalable and reliable communication framework able to work in an heterogeneous and dynamic environment. In this position paper, we propose a generic framework, based on Web Services for interoperability and epidemic or gossip based communication protocols for reliability and scalability that can serve a general management substrate where several Smart Grid problems can be solved. We illustrate the flexibility of the proposed framework by showing how it can be used in two specific scenarios.

I. INTRODUCTION

Electrical power grids are the result of more than 100 years of evolution of a networked infrastructure that has become essential for the regular operation of our society. The grid kept evolving to meet the continuously growing demand from a diversity of consumers, such as industrial, governmental, commercial and residential facilities.

Distributed Energy Resources (DER) embody the future of the power grid because they enable several benefits, such as the reduction of carbon emissions and fuel costs, transmission losses, increased reliability to power failures and deferral of investments among others. More specifically, Distributed Energy Storage (DES) systems are becoming widespread in power grids, mostly combined with intermittent Renewable Power Generators (RPG), as they can be deployed in any stage of a power grid (generation, transmission or distribution) and can also perform various roles according to their features and deployment scenario, and due to the increasing maturity of the technologies and, consequently, of the Return On Investment (ROI)[1], [2]. DES were initially adopted by power utilities for peak shaving, but their adoption is increasing due to their greater cost-effectiveness both from a financial and technical viewpoint, due to the complementarity of RPG and their ability to energize islands, i.e. sections of the grid that are isolated when there is a power outage, improving reliability and reducing service interruptions[3]. This scenario lead to the creation of the microgrid concept, where a small segment of the power grid can still operate while disconnected from the main grid by consuming power drawn from DER. It also paves the way for Combined Heat Production (CHP) which exploits the synergy and complementarity between heat and electricity production, specially in the reduction of carbon emissions and fuel costs, as heat from electrical generation can be more efficiently distributed to nearby customers[4]. The increasing trend in the adoption of DER implies a migration from the current scenario with centralized control and uni-directional power distribution, towards a scenario with distributed and coordinated control and bi-directional power flow. The Smart Grid is thus at the intertwining crisscross of Information and Communication Technologies (ICT) and the legacy power grid. To achieve the complete integration of the various systems that compose the Smart Grid, a general solution will need to, among other requirements, achieve interoperability between largely disparate devices, be scalable, in order to cope with the continuously increasing number of devices in the grid, and be highly reliable due to the criticality of the Smart Grid infrastructure.

The Devices Profile for Web Services (DPWS)[5], which can be considered as the enabler of the Smart Grid[6], provides several of the required features for an Energy SOA, by supporting dynamic, adaptive and auto-configurable architectures, and by embracing the heterogeneity on this environment, achieving full interoperability with Energy systems based on IEC standards[7] as well as with systems that are not directly related with Energy, like a CRM or a ERP. The way gossip protocols work, i.e. disseminating messages in the same way epidemic diseases spread in a population, means they are able to overcome scalability and reliability issues, providing a behavior that can be tuned according to each scenario's requirements. Hence, we propose WS-Gossip, a framework that provides gossip based dissemination, built on top of Web Services, more precisely DPWS.

The rest of this paper is organized as follows: SectionII describes the features of gossip protocols. SectionIII describes relevant Web Services standards included in the DPWS the features of gossip protocols. In sectionIV, the components of the proposed framework are described, as well as possible application scenarios. SectionV refers to significant work related with the proposed approach. SectionVI points some of the benefits of the proposed approach as well as hints for future work.

II. GOSSIP BACKGROUND

In computer networking, gossiping describes the process where a participant that intends to disseminate some information chooses a small random subset of other participants and forwards the information to them. Each of these destinations, upon receiving the information, repeats the same procedure, hence, the gossip moniker. This also mimics also how epidemics spread in populations and, therefore, are also known as epidemic protocols[8]. Despite this simplicity, gossip protocols are highly reliable and scalable and can be tuned to a wide range of performance tradeoffs. In the rest of this section we provide a brief background of gossip protocols and their basic properties.

1) Membership Management: A key component of a gossip protocol is the ability to obtain random subsets of participants to direct messages at in each gossip operation. This component has to provide an uniform random sample and, as much as possible, drawn from a current view of operational, i.e. correct, participants[9]. The first option is to share the full list of participants, allowing each of them to locally draw subsets as desired[10]. This is adequate when the membership does not change frequently, to avoid taxing the network with constant updates, and is small enough to fit each participants memory.

If these conditions are not met, it has also been shown that sufficiently good random samples can be obtained by having each participant keep a small partial view of the system, which is itself maintained using a gossip protocol[11], [8]. A particularly simple but effective approach[12] is allowing a node to exchange some elements in its local list with the same number of elements from some other node. This progressively shuffles the list of each participant and leads to an aproximation of an uniform random sample. By adding a time-based lease and renewal mechanism, it also deals with participants entering and leaving the system.

2) Reliability and Scale: Most interestingly, gossip protocols don't need a reactive mechanism to deal with failures, namely, buffering, acknowledgement, retransmission, and garbage collection, which account for most of the complexity in common communication protocols. Instead, reliability is proactively achieved by the protocol's inherent redundancy and randomization, that cope with both process and network link failures.

The expected probability for a message being delivered to all destinations as a whole can be derived directly from protocol parameters *f*, the number of targets that are locally selected by each process for gossiping, and *r*, maximum number of times a message is relayed before being ignored. By adjusting *r* and *f* parameters according to the expected system size and fault patterns, gossip can be configured such that messages are received with an arbitrary large probability. The key to scalability is that the required fanout configuration is logarithmically proportional to system size. Moreover, because all participants is involved in the dissemination process, the load is evenly spread among everyone.

3) Performance: There are two main variants of gossip protocols[13], which provide different message exchange patterns and performance trade-offs. In *push gossip*, a node that becomes aware of new information, conveys it immediately to target nodes. This variant is adequate for one-to-many dissemination of small messages and events. With *pull gossip*, instead of gossiping upon arrival of new information, a node periodically selects a number of peers and asks them for new information. It has been shown that combining *push* and *pull gossip* results in dissemination being achieved in a lower number of steps[13] and provides a generic framework for gossiping that can be tailored for multiple purposes by parameterizing it with different aggregation functions[14].

In addition, lazily deferring the transmission of payload improves performance in heterogeneous networks, allowing gossip protocols to approximate ideal resource usage efficiency[15]. Such *lazy* variants are most useful when the data payload is very large, but also when it is very likely that the data is already known throughout the network.

Finally, there are two options regarding relaying duplicate messages. In the *infect-and-die* model, a participant that receives the message (i.e. is infected), sends the received message to other nodes, and never sends it afterwards, becoming dead in the analogy with epidemics. In the *infect-forever* model, also known as *balls-and-bins*[16], a participant might relay received message multiple times, possibly until *r* rounds are reached. This last alternative has the advantage of requiring no state at participants to recall recently relayed messages. On the other hand, it usually requires more network resources as the relay limit has to be set conservatively.

III. WEB SERVICES BACKGROUND

The WS-Eventing specification[17] defines the usage of the publish/subscribe pattern by Web Services, and it embodies a flexible filtering mechanism, favoring lightweight implementations and one-to-many communication. It has therefore been the preferred choice for connected devices, namely, within standards like WS-Management[18] and Devices Profile for Web Services (DPWS)[5]. This eventing specification can be combined with other standards, such as WS-ReliableMessaging (WS-RM)[19] for end-to-end acknowledged message delivery or WS-AtomicTransaction (WSAT)[20] for multi-party transactional atomicity guarantees.

DPWS defines a set of protocols that resource constrained devices should implement in order to achieve seamless networking and interoperability through Web Services. It assumes that each device behaves as a standard *hosting service*, providing basal functionality, and exposing one or more *hosted services* that offer device specific functionality. Besides basic SOAP, WSDL, the HTTP binding, WS-Addressing, and WSSecurity that are at the core of Web Services capabilities and interoperability, DPWS also includes WS-Eventing, as previously mentioned, SOAP-over-UDP[21], which enables the usage of UDP as a transport for SOAP messages and enables network level multicast, thus paving the way for dynamic discovery, enabled by combining WS-Discovery[22], WSMetadataExchange[23], and WS-Policy[24]. These protocols allow a client to discover devices in the network, and to learn about their services, resources, characteristics.

Although DPWS provides an adequate infrastructure for small scale systems, such as home automation, it is becoming increasingly interesting when managing large number of components, albeit it has some scale limitations. First, the use of WS-Eventing imposes a burden on the publisher, that has to

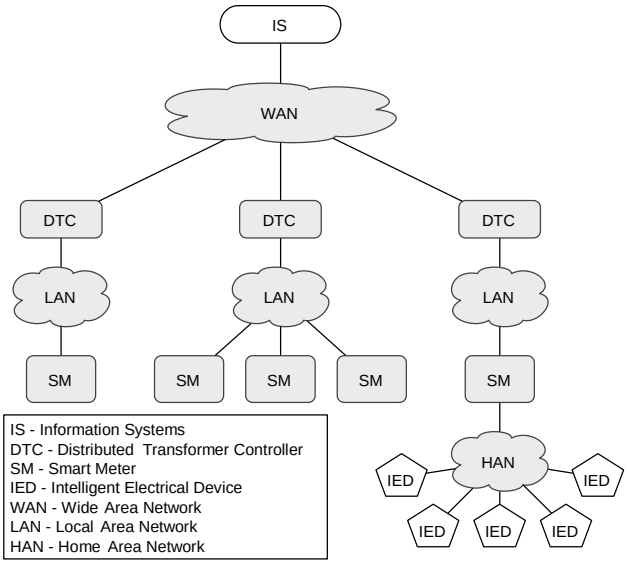

Fig. 1. Overview of a simplified Smart Grid architecture.

notify all subscribers. Moreover, when a resource exposed by many devices has to be updated, e.g. to change a configuration variable, the initiator device must contact every destination individually. Finally, as there is no support for transactional coordination mechanisms, such lengthy operations involving large numbers of destinations are susceptible to faults and cannot be restarted or recovered if stopped. This is particularly worrisome as such notifications and configuration updates may correspond to critical alerts and urgent commands. It does not make sense to resort to heavyweight coordination protocols such as WS-Coordination[27] and WS-AT in such a scenario, because, even if devices could support their requirements, they would not scale to hundreds of participants. Thus, a scalable lightweight coordination protocol, that fits the general DPWS assumptions, is necessary.

IV. PROPOSAL

To exemplify the usage of the proposed framework, we consider a simplified architecture, focusing on the communications infrastructure, of a Smart Grid, as depicted in figure1, which will be described next. The Information Systems (IS) of the utility are the main point for controlling and monitoring the entire smart grid, by retrieving data and issuing commands to other devices in the grid, such as the Distribution Transformer Controllers (DTC), normally connected to a Wide Area Network (WAN). A DTC is installed in a transformer, and it is equipped with sensors and actuators for monitoring the grid's conditions and to enable remote control. As previously mentioned, a DTC interacts with the IS, normally to report retrieved metrics or anomalies on the grid, and with the Smart Meters (SM), connected to the same Local Area Network (LAN), to notify them on tariff changes or service perturbations. A Smart Meter interfaces with the customer, as well as with its appliances or IED through the Home Area Network (HAN) to convey relevant information such as metering, maintenance warnings, among other notifications.

Different types of data and scenarios inside a Smart Grid have different requirements, namely in terms of maximum allowed communications latency. While the transmission of meter readings and market pricing info can tolerate delays ranging from minutes to hours while allowing the loss of some messages, protective relaying, status monitoring and substation SCADA communications endure latencies up to 4 milliseconds, a second, or a few seconds, but the loss of messages of these types might not be well tolerated due to their criticality to the Smart Grid's operation[28], [29]. Gossip protocols can be of particular importance in such settings which are stricter in terms of message delivery assurance compared to message latency, as the message delivery assurance of these protocols largely outweighs the overhead of the additional traffic.

Our proposal to address the scalability and reliability challenges raised by the heterogeneity of the components of the Smart Grid, and its complex nature, is to use WS-Gossip, a Web Services framework for gossip-based dissemination[30]. Gossiping is inherently scalable, as it spreads the load across participants. Moreover, it is also inherently robust, tolerating message loss and participant crashes. This should have the increased advantage of allowing the usage of SOAP-over-UDP even if reliable delivery is desired, which is much less resource consuming than a full fledged HTTP binding over TCP. Moreover, by assuming the Web Services infrastructure based on the Devices Profile for Web Services (DPWS), which allows its usage even in resource constrained devices, we take advantage of each gossiped unit of data being a SOAP envelope, of the selfdocumenting nature of services through WSDL, and of useful base protocols and standards such as WS-Discovery and WSPolicy. Hence, the proposed framework builds upon DPWS to promote interoperability among largely heterogeneous devices, from top of the range mainframes to small IED running completely different operating systems, and it is composed by two services, gossip and peer, that complement each other's functioning. The Gossip Service relies on epidemic protocols for disseminating messages, whereas the Peer Service provides information on the services and devices that are currently on the network. This information can then be used to build and enforce logical overlays on top of the Smart Grid's components, in order to guarantee communication among all of them. For instance, Gossip Service instances rely on this service to obtain the list of targets for disseminating messages.

The usage of the proposed framework is illustrated in two specific scenarios: propagation of relevant information and metrics gathering. On the first scenario, assuming a dynamic tariff, where energy overproduction can lead to significant reduction of prices, this variations must be advertised to all the clients in order to adapt energy consumption accordingly thus stabilizing the network by better matching the demand to the supply of energy. On the second scenario, to better plan future power production, each consumer's smart meter can announce the energy requirements of the connected IED for a specific time frame, and this information will then be aggregated from level to level until reaching the utility's IS. The first scenario focuses on the scalable dissemination capabilities of the framework, whereas the second one demonstrates its data aggregation capabilities.

A. Gossip Service

The Gossip Service provides the ability of epidemic propagation of messages that can include simple data or more com-

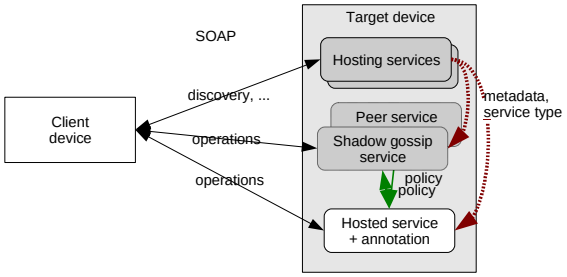

Fig. 2. Overview of WS-Gossip architecture.

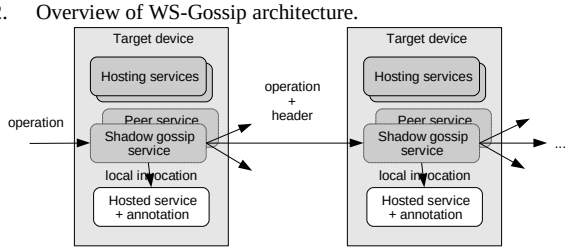

Fig. 3. Initialization of Gossip dissemination.

plex information that was aggregated throughout a network. The general architecture of the proposed gossip service is outlined in Figure IV-A and works as follows. A manufacturer that intends to provide gossip dissemination in its devices can use a DPWS stack with gossiping support and annotate every service supporting gossip using WS-Policy[24] assertions. As a consequence, a shadow gossip service is created for each service where gossip is enabled. Moreover a Peer Service can be setup to provide an entry point to the set of target peers. Multiple shadow gossip services can be attached to the same Peer Service, if they have the same set of targets. Both the original hosted service and its shadow gossip service are advertised to clients that can use each of them independently. A gossip-aware client can examine policy annotations in both these services and determine their relationship. A client may still address the original hosted service, thus maintaining compatibility with existing clients that are unaware of gossiping.

Assume for now a one-way or notification operation (i.e. input or output only)[31], [32] and push-style gossip[13]. Gossiping is started when a client sends a SOAP message to a port in the shadow gossip service, which, upon reception, is inspected to determine if it contains a WS-Gossip header. If not, default gossiping parameters are obtained, including gossip variant, fanout, peer scope or type (according to WSDiscovery), and target binding (HTTP or UDP). Gossiping is then initiated by adding the gossip information to the message header and relaying it to a number of peers and to the local hosted service. This is outlined in Figure IV-A. Upon reception of the gossip message, the dissemination proceeds by each recipient decrementing the message's hop count and forwarding it to its selected peers. Note that a gossip message can be generated by a target device, as depicted in Figure IV-A, but it can also be generated directly by a gossiping-aware client. This allows a client to initiate gossiping with custom parameters to achieve its own reliability and scalability trade-offs.

Since SOAP and WSDL support several operation styles[33], [31], [32], WS-Gossip enables the usage of inputonly operations (i.e. one-way), output-only operations (i.e. notification), and call-back operations (i.e. solicit-response), besides the more typical client-server interaction (i.e. requestresponse). It is also possible that a two-way operation leads to multiple replies. These different operation styles are combined in different gossip variants in addition to the previously described eager push-style, such as the lazy and the pull variants. In request-reply and solicit-response operations, the message is propagated and then all the received replies are propagated back to the initiator. Consider the following example: A request-response to query the last measured voltage value throughout a segment of a Smart Grid. The client invokes the operation on the shadow service, which is forwarded to its known peers and eventually reaches all targets, i.e. all the IED in that Smart Grid's segment. Along the way, each of these peers will decrement the value of the hops parameter in the message and forward it to their own set of peers, previously obtained from the Peer Service, or retrieve this information in the event that the device still does not possess such information or if it is obsolete according to the configuration parameters. These operations repeat time and time again, normally until the hops parameter on the message has been decremented to zero. All responses then travel back along the same tree implicitly created by the request message, eventually reaching the initiator.

An alternative is to use a filter, which can omit or aggregate replies according to some rule specified when gossip is initiated. Consider the following example: The determination of the maximum voltage phase angle difference in a given segment of a Smart Grid. Assuming an IED on the grid, it can start this process by gossiping a message containing an XSLT definition of the aggregation function to be applied by each node to combine its own data with the aggregated one in the received message. Assuming a request-response aggregation invocation, after reaching all targets, responses will then travel back along the same tree implicitly created by the request message, but they are buffered and filtered using the conveyed aggregation function, such that only one aggregated value is returned by each peer. The reply is sent by each peer as soon as a configured minimum of targets have replied, conveying a value, or a fault, for instance, when the timeout expires or in the occurrence of other errors.

B. Peer Service

The Peer Service acquires and manages information on the devices and services that are available in the network, which can be queried by clients for their purposes, but more specifically in the case of the Gossip Service, it should provide information on possible targets for disseminating messages, allowing the definition of overlays. Various instances of this service running in disparate devices and places of a network can cooperate and exchange information in order to build a more complete image of the devices and services currently available in a network. These maintenance communications can be performed by using the Gossip Service.

By default, the Gossip Service does not need an explicit peer management service. Instead, each gossip interaction is configured with a service type that can be used to discover the full set of reachable peers through WS-Discovery. This is

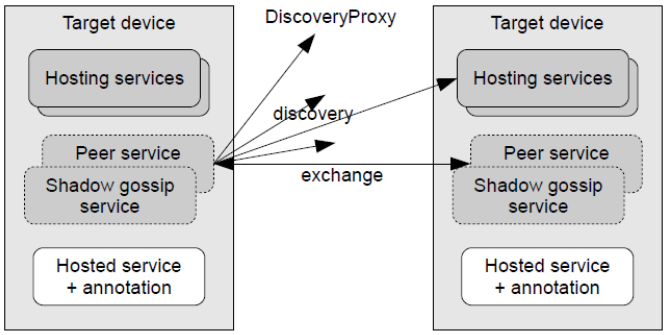

Fig. 4. Overview of peer management.

most useful in scenarios where a discovery proxy device exists: In this case, when a set of peers is required for gossiping, it can be obtained efficiently by querying the proxy. This leads to a configuration with centralized peer information while information dissemination is distributed, which is adequate for scenarios with low churn and relatively high messaging rate.

This service leverages WS-Discovery by exploiting its AdHoc mode, specially in networks with a large churn, to continuously update the overlay through the inspection of multicast messages which announce when a device has started, stopped or been modified. In stable networks, the Membership Service can be coupled with a Discovery Proxy, hence accessing the same information this component possesses due to direct notifications of devices entering and exiting the network.

If a proxy is not available, the usage of the probe or resolution mechanisms of WS-Discovery in Ad-Hoc mode would lead to a large number of multicast messages which would most likely defeat the purpose of gossip. Instead, our proposal allows discovered peers to be cached locally and exchanged with other peers to implicitly create an overlay network using the Newscast protocol[14]. The structure of the stored peer information comprises a list where each service instance is represented by an entry that contains its endpoint address, type and the identification of the device that hosts the service. Additionally, each entry keeps an heartbeat counter that is incremented as exchanged update messages contain information on this service. This information is exchanged among different devices and also updated through the examination of WS-Discovery multicast messages issued by devices and services entering or leaving the network. Periodically, if the instance has not received a request for exchanging its membership information during a certain time frame, it selects another instance of the peer service to which it sends such a request containing the list of the known endpoints. Upon reception of such a message, the contacted instance returns to the requester its own list of known endpoints, and merges it with the received one.

The heartbeat counter of a service instance that never sends a new message, or eventually sends but without reaching any peer service instance, stays unchanged, implying that it will move towards the end of the membership list as the counter of other services is being updated and new services are discovered. That service instance will eventually be discarded when the cache of the Peer Service reaches the configured maximum size.

C. Solution

Regarding the multitude of Smart Grid scenarios, where largely heterogeneous devices interact, we propose the usage of gossip based dissemination to replace the existing mechanisms of alerts and events propagation for scenarios with an high rate of messages, a large number of targets, since publishers might be overwhelmed with the subscriptions storage and maintenance, and important or critical messages whose loss is badly tolerated by the systems.

It has been shown that DPWS is suitable for smart meters communication[34], but for a large amount of devices, in the region of some thousands, an hierarchically structured communication does not cope with the generated traffic[35]. To circumvent this limitation, we propose the usage of the WS-Gossip framework, namely in two scenarios related to the Automated Metering Infrastructure (AMI): propagation of relevant information and metrics gathering.

In the event of unexpected renewables overproduction, as occurs in wind farms due to a sudden wind pickup, the overproduced energy must be consumed to maintain the grid stable and working correctly. When dealing with such an event, an utility can store this energy using DES, but it can also sell the excess energy with more attractive rates than the normal tariffs to its customers[36]. The AMI comprises two-way communication between the utility's systems and Smart Meters, allowing the conveyance of information in both directions. These tariff modifications will then flow from the utility's IS to all the customers through their own smart meters or even through some other IED. We will focus how these communications can occur using our framework in such a scenario. When the IS become aware of such an overproduction scenario and decide to set lower tariffs, this information is then encapsulated in a push gossip message which is disseminated to the target DTCs retrieved from the Peer Service deployed at the IS node. The Gossip Service instance of the targets, upon reception of the message, decrements the number of hops and retransmits the message to the target nodes that its Peer Service instance proposes, which could be other DTCs, reachable through the WAN, or Smart Meters, reachable through the LAN to which the sending DTC is connected. When a Smart Meter receives the message, the Gossip Service instance behaves in a similar fashion to the one in the DTCs, i.e., it retransmits it to the targets that the Peer Service instance provides. This instance can be located at the Smart Meter or at any other reachable node. The targets can vary from other Smart Meters, connected to the same LAN, to IED connected to the same HAN, that can range from appliances to RPG. IED can adapt their operating mode according to the received information of reduced tariffs. For instance, by analyzing the price reduction and the corresponding period, HVAC can increase its consumption to better suit the consumer's temperature preferences, dishwashers and washing machines can start their washing cycles before scheduled, among other possibilities. In parallel with the retransmission of the message to the designated targets, the Gossip Service instance of the Smart Meter decapsulates the notification on tariffs reduction from the message, which it then presents in a local display or forwards it to some other device, as configured

by the customer, like a smartphone or a tablet. To summarize the dissemination behavior of the framework in this scenario, notice that:

- A Gossip Service instance will forward the encapsulated message to the specified action of the targeted service if an instance of that service coexists in the same node.
- Any Gossip Service instance running at any node identified on figure1, will only retransmit the message on its first reception, following the *infect-and-die* model, and if the number of hops is greater than zero as well.
- The targets for each dissemination are retrieved from an instance of the Peer Service that could be or not co-located with the sending instance of the Gossip Service which received the message previously.

In the second scenario, the proposed framework is used to collect metrics from different points of the Smart Grid in order to plan power production according to the announced energy requirements. For instance, electric vehicles charging, dishwashing or clothes washing are performed at night, when tariffs usually are lower. Periodically, the central IS invokes the Pull Aggregation operation on DTCs, which, in their turn, invoke the same operation on the target SMs designated by their Peer Service. A Smart Meter, upon reception of such a request, propagates the same request to the IED pointed by the Peer Service in the HAN. Each of these IED, if configured to perform a night period task, such as the ones previously mentioned, will respond with the energy requirements to execute the configured tasks, the duration of the tasks, and the time by which the tasks should be finished. The Smart Meter will then aggregate this information for the entire household, after waiting for responses from IED until a certain number of responses arrives or a certain timeout elapses, according to configured preferences. The aggregate information will combine all the power requirements pointed by the IED for the three 8 hour time periods which divide the day. For simplification purposes, we will consider that all the energy needs for each of these periods will be simply added in order to produce the aggregate information at the Smart Meters. Each DTC will then receive the aggregate responses from the previously contacted Smart Meters, and again, having waited according to configured preferences, these responses will be aggregated into a response sent to the IS. The IS will then process this message and assess what are the announced energy requirements and plan the energy generation according to the demand for the upcoming night.

V. Related Work

WS-SCADA[37] attempts to address integration needs of clients, applications, utilities and market participants, by accommodating information needs of all participants and adapting to dynamic changes at both system and business level. The proposed open, flexible and scalable infrastructure for information integration includes two Web Services protocol stacks, for a control center and a substation, and both share with DPWS a lot of similarities in their composition. For instance, the WS-Discovery protocol, combined with multicast SOAPover-UDP communication, provides Plug-and-Play features for IEDs and allows substations to locate IEDs and respective services. The WS-Eventing protocol can be used by a control center to notify substations on control messages, and also to be notified on status information and realtime operation data of substations, such as voltage, current, breaker status, and phasor measurement data. A substation can also subscribe to notifications from IEDs, which may be redundant, if regarding measurements of a single signal. These redundancies must be avoided prior to issuing the notifications, at the IEDs' level, or afterwards by the substation.

Regarding communications among devices in a LAN, an extension for the usage of UDP Multicast with WS-Eventing was proposed[38], which could help reduce the amount of traffic in scenarios where a single publisher must inform various subscribers on the occurrence of periodic events. However, the assurance of reliable delivery of events using a positive acknowledgment system would cause an acknowledgment explosion in the publisher. And even in the case of well-known periodic events, where the usage of notification retransmission requests would be suitable, it could lead to a similar scenario if various subscribers do not receive the same event, since each will trigger a retransmission request which will ultimately accumulate on the publisher's side.

VI. Conclusion

The implementation of the Smart Grid is highly supported by Information and Communication Technologies (ICT), through interconnecting platforms that must integrate different technologies, which are present in the multitude of systems composing current and future power grids. For such a platform, we propose a framework based on SOA, and, more specifically, on the Devices Profile for Web Services (DPWS), which will allow, for instance, communications based on web services between resource constrained devices and mainframes, automatic detection of the devices present on the network, easy integration of new devices. The adoption of gossip based communication variants will provide probabilistic message delivery guarantees as well as proactive reliability, which can be set through the values of the gossip parameters, showing its adaptability to disparate timeliness and message loss requirements of different systems composing the Smart Grid.

The future work will consist on the assessment of the various dissemination and aggregation variants provided by the described framework, namely, evaluating its performance in Smart Grid related scenarios in comparison with other similar systems.

References

[1] V. Gungor, D. Sahin, T. Kocak, S. Ergut, and C. Buccella..., "A survey on smart grid potential applications and communication requirements," *ieeexplore.ieee.org*, Sep 2013.

[2] C. Nguyen and A. Flueck, "Agent based restoration with distributed energy storage support in smart grids," *Smart Grid*, Jan 2012.

[3] A. Nourai and D. Kearns, "Batteries included," *Power and Energy Magazine, IEEE*, vol. 8, no. 2, pp. 49–54, 2010.

[4] A. K. Basu, S. P. Chowdhury, S. Chowdhury, and S. Paul, "Microgrids: Energy management by strategic deployment of ders—a comprehensive survey," *Renewable and Sustainable Energy Reviews*, vol. 15, no. 9, pp. 4348–4356, Dec 2011.

[5] "Devices Profile for Web Services (DPWS) 1.1 OASIS Standard," http://docs.oasis-open.org/ws-dd/dpws/1.1/os/wsdd-dpws-1.1-specos.html, pp. 1–43, 01 July 2009.

[6] S. Karnouskos, “Asset monitoring in the service-oriented internet of things empowered smartgrid,” *Service Oriented Computing and Applications*, Jan 2012. [Online]. Available: http://www.springerlink.com/index/J2H1258618305335.pdf

[7] J. Schmutzler, S. Groning, and C. Wietfeld, “Management of distributed energy resources in iec 61850 using web services on devices,” pp. 315–320, 2011.

[8] P. Eugster, R. Guerraoui, A. Kermarrec, and L. Massoulie, “Epidemic information dissemination in distributed systems,” *Computer*, vol. 37, no. 5, pp. 60 – 67, 2004.

[9] M. Jelasity, R. Guerraoui, A.-M. Kermarrec, and M. van Steen, “The Peer Sampling Service: Experimental Evaluation of Unstructured Gossip-Based Implementations,” in *Middleware ’04: Proceedings of the 5th ACM/IFIP/USENIX international conference on Middleware*. New York, NY, USA: Springer-Verlag New York, Inc., 2004, pp. 79–98.

[10] K. P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky, “Bimodal multicast,” *ACM Trans. Comput. Syst.*, vol. 17, no. 2, pp. 41–88, 1999.

[11] P. T. Eugster, R. Guerraoui, S. B. Handurukande, P. Kouznetsov, and A.-M. Kermarrec, “Lightweight Probabilistic Broadcast,” *ACM Trans. Comput. Syst.*, vol. 21, no. 4, pp. 341–374, 2003.

[12] S. Voulgaris, D. Gavidia, and M. van Steen, “Cyclon: Inexpensive membership management for unstructured p2p overlays,” *Journal of Network and Systems Management*, vol. 13, no. 2, pp. 1–21, June 2005.

[13] R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking, “Randomized rumor spreading,” *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pp. 565–574, 2000.

[14] M. Jelasity, W. Kowalczyk, and M. van Steen, “Newscast computing,” Vrije Universiteit Amsterdam, Department of Computer Science, Amsterdam, The Netherlands, Tech. Rep. IR-CS-006, Nov. 2003.

[15] J. Pereira, R. Oliveira, and L. Rodrigues, “Efficient epidemic multicast in heterogeneous networks,” in *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops*, vol. 4278/2006. Springer Berlin / Heidelberg, October 2006, pp. 1520–1529.

[16] B. Koldehofe, “Simple gossiping with balls and bins,” in *Proceedings of the 6th International Conference on Principles of Distributed Systems (OPODIS’02)*, 2002, pp. 109–118.

[17] “WS-Eventing W3C Member Submission,” http://www.w3.org/Submission/WS-Eventing/, 15 March 2006.

[18] “Web Services for Management (WSManagement) 1.1 DMTF Standard,” http://www.dmtf.org/standards/published documents/DSP0226 1.1.pdf, 31 March 2010.

[19] “WS-ReliableMessaging 1.2 OASIS Standard,” http://docs.oasisopen.org/ws-rx/wsrm/200702/wsrm-1.2-spec-os.html, 02 February 2009.

[20] “Web Services Atomic Transaction (WS-AT) 1.2 OASIS Standard,” http://docs.oasis-open.org/ws-tx/wstx-wsat-1.2-spec-os/wstx-wsat-1.2spec-os.html, pp. 1–26, 02 February 2009.

[21] “SOAP-over-UDP Version 1.1 OASIS Standard,” http://docs.oasisopen.org/ws-dd/soapoverudp/1.1/os/wsdd-soapoverudp-1.1-specos.html, 01 July 2009.

[22] “Web Services Dynamic Discovery (WS-Discovery) 1.1 OASIS Standard,” http://docs.oasis-open.org/ws-dd/discovery/1.1/os/wsdddiscovery-1.1-spec-os.html, pp. 1–50, 01 July 2009.

[23] “Web Services Metadata Exchange 1.1 (WS-MetadataExchange) W3C Member Submission,” http://www.w3.org/Submission/2008/SUBMWS-MetadataExchange-20080813/, 13 August 2008.

[24] “Web Services Policy (WS-Policy) 1.5 - Framework,” http://www.w3.org/TR/ws-policy/, 2007.

[25] “WS4D (Web Services for Devices),” http://www.ws4d.org/, 20 February 2011.

[26] “SOA4D (Service-Oriented Architecture for Devices),” https://forge.soa4d.org/, 20 February 2011.

[27] “Web Services Coordination (WS-Coordination) 1.2 OASIS Standard,” http://docs.oasis-open.org/ws-tx/wstx-wscoor-1.2-spec-os/wstxwscoor-1.2-spec-os.html, pp. 1–26, 02 February 2009.

[28] D. Li, Z. Aung, J. Williams, and A. Sanchez, “Efficient authentication scheme for data aggregation in smart grid with fault tolerance and fault diagnosis,” pp. 1–8, 2012.

[29] D. Rua, L. Pereira, and N. Gil..., “Impact of multi-microgrid communication systems in islanded operation,” *Innovative Smart Grid ...*, Jan 2011.

[30] *SIGOPS Oper. Syst. Rev.*, vol. 41, no. 5, 2007.

[31] (2001) Web Services Description Language (WSDL) 1.1. http://www.w3.org/TR/wsdl.

[32] (2003) Web Services Description Language (WSDL) 1.2. http://www.w3.org/TR/2003/WD-wsdl12-20030303/.

[33] (2007, 23 April) SOAP Version 1.2 Part 0: Primer (Second Edition) W3C Recommendation. http://www.w3.org/TR/2007/RECsoap12-part0-20070427/.

[34] S. Karnouskos and A. Izmaylova, “Simulation of web service enabled smart meters in an event-based infrastructure,” *Industrial Informatics, 2009. INDIN 2009. 7th IEEE International Conference on*, pp. 125 – 130, 2009.

[35] S. Karnouskos, P. G. da Silva, and D. Ilic, “Assessment of highperformance smart metering for the web service enabled smart grid era,” *Proceeding of the second joint WOSP/SIPEW international conference on Performance engineering*, pp. 133–144, 2011.

[36] M. Morgan, J. Apt, L. Lave, M. Ilic, M. Sirbu, and J. Peha, “The many meanings of” smart grid”,” *repository.cmu.edu*, Jan 2009.

[37] Q. Chen, H. Ghenniwa, and W. Shen, “Web-services infrastructure for information integration in power systems,” *Power Engineering Society ...*, Jan 2006.

[38] D. Gregorczyk, “Ws-eventing soap-over-udp multicast extension,” *Web Services (ICWS), 2011 IEEE International Conference on*, pp. 660 – 665, 2011.