A resilient and scalable service architecture for the Smart Grid

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

David Rua
Power Systems Unit
INESC TEC & University of Porto,
Porto, Portugal
Email: drua@inescporto.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 requires 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

Smart Grids (SG) have been introducing a paradigm change in electric power system with the objective of enhancing the integration of renewables and promote the generalized participation of different entities. Unprecedented research initiatives have been established with the purpose of addressing the architectural and technological aspects of power, information and communications systems [**?**], [**?**]. The SG concept includes different visions and strategies that allow the modernization of the electric industry in order to ensure high levels of robustness, adaptability, scalability, security, economy, self-healing and protection in highly dynamic systems [**?**].

SG embody the future of the power grid because of the associated benefits, such as the reduction of carbon emissions and fuel costs, transmission losses, increased reliability to power failures and deferral of investments among others. Distributed Energy Resources and Distributed Energy Storage systems are becoming widespread in power grids, combined with intermittent Renewable Energy Sources, in different segments of the power system requiring monitoring and control schemes to allow their enhanced participation in both market and system services.

The role of Information and Communication Technologies (ICT) in SG is gaining importance since it represents the underlying support infrastructure that allow the necessary information exchange towards the integration of different participants while supporting a diversified set of applications and services. As such, a complex interconnection between different segments, domains, and players requires a suitable ICT. To achieve the complete integration of the various systems that compose the SG, a general ICT solution will need, among others, to: achieve the necessary 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 to support the operational requirements introduced by the SG.

The Devices Profile for Web Services (DPWS) [**?**], can have an important role in SG [**?**], as it 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, thus achieving full interoperability with energy systems based on the main standards and models [**?**] as well as with other systems. Gossip protocols are potentially able to overcome scalability and reliability issues, providing a behavior that can be tuned according to each scenario's requirements. They can be used in SG with different aims, namely for secondary and tertiary control on a microgrid [**?**], or for disseminating and aggregating important information in the Automated Metering Infrastructure [**?**]. We thus propose WS-Gossip, a framework that provides gossip based dissemination, built on top of Web Services, more precisely DPWS within the SG context.

The rest of this paper is organized as follows: Section II and Section III provide background on gossip protocols and Web Services standards, respectively. Section IV describes the components of the proposed approach and some application scenarios. Section V presents related work and Section VI concludes the paper and provides directions 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 how epidemics spread in populations, justifying the alternative denomination of epidemic protocols [**?**]. Despite simple, gossip protocols are highly reliable, scalable and adjustable 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: A key component of a gossip protocol is the ability to obtain random subsets of participants to communicate with. This component has to provide a uniform random sample of operational, i.e. correct, participants [**?**]. A naive approach is to share the full list of participants, allowing each of them to locally draw subsets as desired [**?**], which is adequate when the membership does not change frequently and is small enough to fit in memory. Alternatively, it has been shown that good random samples can be obtained by having each participant keep a small partial view of the system maintained using a gossip protocol [**?**]. A particularly simple but effective approach [**?**] 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 approximated 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: Reliability is proactively achieved by the redundancy and randomization of gossip protocols, coping with both process and network link failures. The expected probability for a message being delivered to all destinations 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 value of f is logarithmically proportional to system size. Moreover, the load is evenly spread among everyone because all participants are involved in the dissemination process.

3) Performance: There are two main variants of message exchange patters in gossip protocols [**?**], which provide different performance trade-offs. In *push gossip*, a node that becomes aware of new information, conveys it immediately to target nodes, which is adequate for one-to-many dissemination of small messages and events. With *pull gossip*, a node periodically selects a number of peers and asks them for new information. Combining *push* and *pull gossip* results in dissemination being achieved in a lower number of steps [**?**] and provides a generic framework for gossiping that can be tailored for multiple purposes by parameterizing it with different aggregation functions. In addition, lazily deferring the transmission of payload improves performance in heterogeneous networks, allowing gossip protocols to approximate ideal resource usage efficiency [**?**].

III. Web Services background

WS-Eventing 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 and Devices Profile for Web Services (DPWS) [**?**]. WS-Eventing can be combined with other standards, such as WS-ReliableMessaging for end-to-end acknowledged message delivery or WS-AtomicTransaction (WS-AT) 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 WS-Security, that are at the core of Web Services capabilities and interoperability, DPWS also includes WS-Eventing, as previously mentioned, SOAP-over-UDP, enabling UDP as a transport for SOAP messages and network level multicast, which paves the way for dynamic discovery, enabled by combining WS-Discovery, WS-MetadataExchange, and WS-Policy.

Although DPWS provides an adequate infrastructure for small scale systems, 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 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 and WS-AT in such a scenario, because, even if devices could support their requirements, they would not scale very well. 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 (SG), as depicted in figure 1, 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 Secondary Substation Controllers (SSC), normally connected to a Wide Area Network (WAN). A SSC is installed in an electric distribution transformer, and it is equipped with sensors and actuators for monitoring the grid's conditions and enable remote control. As previously mentioned, a SSC interacts with the IS, normally to report retrieved metrics or anomalies on the grid, and with the Smart Meters (SM), connected to the same Field Area Network (FAN), to notify them on tariff changes or service perturbations. A Smart Meter interfaces with the customer, as well as with its appliances or Intelligent Electronic Devices (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 SG 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

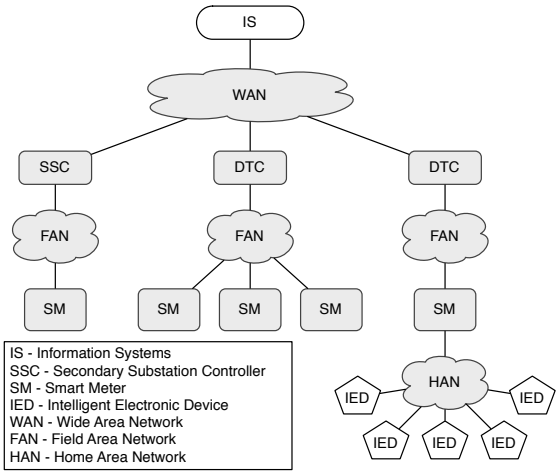

Fig. 1. Overview of a simplified Smart Grid architecture.

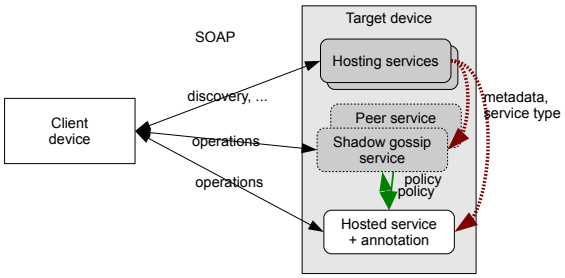

Fig. 2. Overview of WS-Gossip architecture.

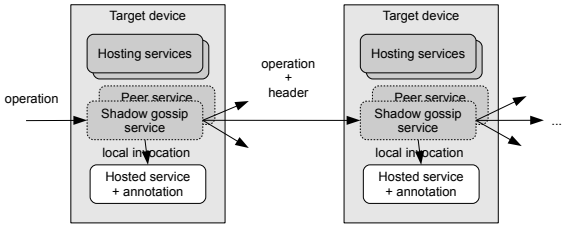

Fig. 3. Initialization of Gossip dissemination.

messages, protective relaying, status monitoring and substation SCADA communications endure latency values as low as a few milliseconds to seconds or even minutes, but the loss of messages of these types might not be well tolerated due to their criticality to the SG operation [**?**], [**?**]. 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 SG, and its complex nature, is to use WS-Gossip, a Web Services framework for gossip-based dissemination. The inherent scalability and reliability of gossip protocols allows the usage of SOAP-over-UDP even if reliable delivery is desired, since it 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), we take advantage of each gossiped unit of data being a SOAP envelope, of the self-documenting nature of services through WSDL, and of useful base protocols and standards such as WS-Discovery and WS-Policy. 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. The Gossip Service relies on gossip 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 SG's components, in order to guarantee communication among all of them. 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, these 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 SM 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 complex information that was aggregated throughout a network. The general architecture of the proposed service is outlined in Figure 2 and works as follows. To provide gossip dissemination in its devices, a manufacturer can use a DPWS stack with gossiping support and annotate every service supporting gossip using WS-Policy 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. 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 legacy clients.

Assume for now a one-way or notification operation (i.e. input or output only) and push-style gossip [**?**]. Gossiping is started when a client sends a SOAP message to a port in the shadow gossip service, which, is then inspected to determine if it contains a WS-Gossip header. If not, default gossiping parameters are obtained, including gossip variant, fanout f, rounds r, peer scope or type (according to WS-Discovery), and target binding (HTTP or UDP). Gossiping is then initiated by adding these parameters to the message header and relaying it to a number of peers and to the local hosted service. Upon

reception of the gossip message, the dissemination proceeds by each recipient decrementing the message's r counter and forwarding it to its selected peers. Note that a gossip message can be generated by a target device, as depicted in Figure 3, or directly by a gossiping-aware client, allowing it to set the gossiping parameters to achieve customized reliability and scalability trade-offs.

WS-Gossip also supports output-only operations (i.e. notification), and call-back operations (i.e. solicit-response), besides the more typical client-server interaction (i.e. request-response). These operation styles are combined in different gossip variants, such as lazy pull, in addition to the previously described eager push-style. 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 SG. 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 SG's segment. Along the way, each of these peers will decrement the message's r value and forward it to their own set of peers, until the message's r value reaches zero, 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 are repeated by recipients, normally until the message's r value reaches 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 SG. 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 most useful in scenarios where a Discovery Proxy device exists, since it can be queried directly to obtain a set of peers for gossiping. 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. The usage of the probe or resolution mechanisms of WS-Discovery in Ad-Hoc mode by the Gossip Service to collect a set of peers would lead to a large number of multicast messages which would most likely defeat the purpose of gossip. On the contrary, the Peer Service exploits this WS-Discovery 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, without resorting to probe or resolution requests. Discovered peers can then be locally cached and exchanged with other peers to implicitly create an overlay network using the Newscast protocol [**?**]. The Peer Service can also 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. 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 or if messages sent by it are detected by the Peer Service. If the heartbeat counter of a service instance remains unchanged for a long time, it will shift towards the end of the membership list as the counter of other services is updated and new services are discovered. That service instance will eventually be discarded when the cache of the Peer Service reaches the maximum configured size. Periodically, each Peer Service instance will select another instance to exchange its list of known endpoints. Upon reception of such a message, the contacted instance returns its own entries list to the requester, and merges it with the received one.

C. Solution

Regarding the multitude of SG 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 [**?**], but for a large amount of devices, in the region of some thousands, an hierarchically structured communication does not cope well with the generated traffic [**?**]. 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.

The use of dynamic price tariff schemes through AMI

systems can allow utilities to take advantage of operational scenarios to shape the participation of customers, by setting more, or less, attractive tariffs to its customers, while guaranteeing a stable and normal operation of the power system [**?**]. The AMI comprises two-way communication between the utility's systems and SMs, 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 SM or even through some other IED. We will focus how these communications can occur using our framework in such a scenario. When a utility decides to set lower price tariff through its IS, this information is then encapsulated in a push gossip message which is disseminated to the target SSCs retrieved from the Peer Service deployed at the IS node. The Gossip Service instance of the targets, upon reception of the message, decrements the value of rounds r and retransmits the message to the target nodes that its Peer Service instance proposes, which could be other SSCs, reachable through the WAN, or SMs, reachable through the FAN to which the sending SSC is connected. When a SM receives the message, the Gossip Service instance behaves in a similar fashion to the one in the SSCs, i.e., it retransmits it to the targets provided by the Peer Service instance. This instance can be located at the Smart Meter or at any other reachable node. The targets can vary from other SMs, connected to the same FAN, to IED connected to the same HAN, that can range from controllable appliances to Renewable Energy Sources (RES). IED can adapt their operating mode according to the received information of the tariff scheme. 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 SM can present the notification on tariffs reduction in a local display or forward 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 forwards the encapsulated message to the specified action of the targeted service if such an instance coexists in the same node. Any Gossip Service instance running at any node identified on figure 1, only retransmits the message if r 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.

In the second scenario, the proposed framework is used to collect metrics from different points of the SG 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 SSCs, which, in their turn, invoke the same operation on the target SMs designated by their Peer Service. A SM, 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 nightly task, will respond with the energy requirements to execute the configured tasks, their duration, and the time by which they should be finished. The SM 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 SMs. Each SSC will then receive the aggregate responses from the previously contacted SMs, and again, having waited according to configured preferences, will aggregate those responses in a single 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 [**?**] 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 provides Plug-and-Play features for Intelligent Electronic Devices (IED) allowing substations to locate them and their services. WS-Eventing can be used by substations to receive notifications from IED, and by a control center to notify substations on control messages or to be notified on status information and real–time operation data of substations, such as voltage, current, breaker status, and phasor measurements.

Regarding communications among devices in a LAN, an extension for the usage of UDP Multicast with WS-Eventing was proposed [**?**], which could help reduce the amount of traffic in scenarios where a single publisher must inform various subscribers on the occurrence of periodic events. It avoids an acknowledgment explosion on the publisher's side, but the usage of notification retransmission requests can lead to a similar explosion on the publisher's side, if various subscribers do not receive the same event.

Gossip protocols can be used in SG to disseminate important information, for instance, load shedding notifications, or metering data aggregation [**?**]. However, limiting epidemic dissemination to a single pairwise interaction per node in each cycle leads large dissemination times. The more traditional approach, where each gossiping node contacts various of its neighbors in parallel would prove to be useful in the majority of the SG scenarios. Secondary and tertiary controls in microgrids can be implemented with gossip protocols with the aim of improving power quality and optimizing generation costs, respectively [**?**]. A gossip aggregation protocol can be used to calculate the average of voltage and frequency deviations measured at Distributed Energy Resources (DER) units, which can then be added to the reference active and reactive power in order to stabilize the network, by decreasing the deviations. To optimize distributed generation costs, each DER unit periodically contacts a random neighbor in order to harmonize their marginal cost functions. Such scenarios could

benefit from more advanced aggregation strategies, in order to decrease the number of communication interactions between the DER units.

VI. Conclusion

The implementation of Smart Grids (SG) is highly supported by Information and Communication Technologies, 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, 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 provides 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 SG.

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