SmartSociety

Hybrid and Diversity-Aware Collective Adaptive Systems
When People Meet Machines to Build a Smarter Society

Grant Agreement No. 600584

Deliverable 8.2 Working Package 8

Platform Prototype: Early Results and System Design

Dissemination Level (Confidentiality):[1]	*CO*
Delivery Date in Annex I:	*31/12/2014*
Actual Delivery Date	*December 12, 2014*
Status[2]	*D*
Total Number of pages:	17
Keywords:	*Platform, Prototype, Integration*

[1]PU: Public; RE: Restricted to Group; PP: Restricted to Programme; CO: Consortium Confidential as specified in the Grant Agreeement
[2]F: Final; D: Draft; RD: Revised Draft

Disclaimer

Full project title:	SmartSociety: Hybrid and Diversity-Aware Collective Adaptive Systems: When People Meet Machines to Build a Smarter Society
Project Acronym:	SmartSociety
Grant Agreement Number:	600854
Number and title of work-package:	8 Architecture and Integration
Document title:	Platform Prototype: Early Results and System Design
Work-package leader:	Iacopo Carreras, UH
Deliverable owner:	Iacopo Carreras, Daniele Miorandi
Quality Assessor:	Luc Moreau, SOTON

List of Contributors

Partner Acronym	Contributor
UH	Daniele Miorandi, Iacopo Carreras, Tommaso Schiavinotto

Executive Summary

TBA

Table of Contents

List of Acronyms

Acronym	Full Name	Description
EC	Evaluator Component	System component in charge of evaluating the outcomes of each computation task (Sec. **??**).
IM	Incentives Manager	System component in charge of managing the implementation of incentive schemes (Sec. **??**).
KB	Knowledge Base	System component in charge of storing and managing the knowledge in the platform (Sec. **??**).
OC	Orchestration and Coordination	System component in charge of coordinating the orchestrating the execution of SmartSociety programs (Sec. **??**).
PDE	Peer Discovery Engine	System component in handling peers querying and discovery (Sec. **??**).
PF	Programming Framework	System component in charge of the interface between the SmartSociety platform and developers (Sec. **??**).
PM	Peer Manager	System component in charge of managing peers (Sec. **??**).
TCO	Task Coordination and Orchestration	System sub-component in charge of coordinating the execution of a single task (Sec. **??**).
TEM	Task Execution Manager	System component in charge of managing the execution of a single task by external peers (Sec. **??**).
WE	Workflow Engine	System component in charge of managing the execution of a program workflow (Sec. **??**).

1 Introduction

This report is a short accompanying document whose aim is to describe the platform components developed and integrated within a coherent platform by the Consortium partners during the second reporting period. The actual deliverable is a prototype of the SmartSociety platform, which can be found at: ...

Starting from the analysis of requirements and initial platform architecture in D8.1, WP8 undertook an intense dialogue with technical workpackages (WP2-WP7) in order to ensure full alignment of the top-level platform architecture and of the scientific and technical outcomes of the single WPs. The results of this first phase was a revised architecture for the SmartSociety platform, which is presented in Sec. 2. According with the progress of the technical WPs, interfaces specifications were defined, which are summarised in Sec. 3. During the second half of the year the actual integration of components started, resulted in a 'minified' version of the platform which is described in Sec. 4. The missing components will be integrated during the third year according to the roadmap outlined in Sec. **??**.

2 The SmartSociety Platform Revised Architecture

The SmartSociety project aims at the design, implementation and validation of a platform to support the execution of hybrid human-machine computation tasks. The external actors interacting with the platform will be (i) developers, who will create applications to be deployed and run over the SmartSociety platform (ii) external peers (both humans and machines) which are expected to be integrated into the platform by means of external applications (iii) system administrator, who is expected to ensure the proper functioning of the platform over time.

The core platform will support developer in the creation of applications which involve the execution of hybrid human-machine computational tasks.

An application deployed over the SmartSociety platform can include:

- a SmartSociety application: this is mandatory and implements the overall logic of the application, including the complete description of the tasks to be executed through SmartSociety by peers and the overall logic of the application. The application will be written in a high-level programming language and will rely on the SmartSociety programming abstractions.

- an external application: this will be written by developers (e.g., a mobile application) and is interfaced with the application running in the SmartSociety runtime. These applications are the interface towards end-users for consuming applications based on SmartSociety.

- peers application: this is a dedicated application that will mediate the interactions with peers, especially with humans and collectives. Examples of such applications include mobile applications or a social network applications (e.g., Facebook application). Through such applications it will be possible to sign up peers into the platform, deliver them computation tasks and eventually collect the outcomes of such executions. In addition, through such applications it will also be possible to exploit external context for a proper matching between tasks and peers.

- an external web interface: this interface can be used to visualize the data collected from the peers' computations, as well as to monitor the proper operation of the application. This interface will be accessible for any application deployed over the platform.

Through a set of pre-defined APIs, external applications can interact with the applications running in the SmartSociety platform, issuing computation tasks and receiving results after their execution. This process will be asynchronous, given the highly variability of peers availability and of the type computations targeted in SmartSociety.

During the execution of the computational tasks, the platform will therefore interact with human peers, machine peers, human collectives and hybrid collectives. As highlighted in Fig. 1, such interactions will be mediated via a dedicated application, which is linked to the SmartSociety application deployed over the platform. Through such application peers will execute computations tasks.
The platform can also interact with external data sources and external services in order to complete the assigned computation. Examples of such data sources are open data from a public repository, or a web service, accessible through some API. The integration of such external services will be supported by means of dedicated APIs to be invoked when running the application.
A web GUI will also be present, which will to access detailed information about the overall status of the platform. This interface will only be accessible to the platform administrators and will provide access to the platform status information, but not to data specific to the applications being deployed over the platform.

In the following sections, we will provide various views on the refined design of the platform. During the second year, we have followed use-case driven approach in order to progress in the refinement of the initial version of the architecture documented in Deliverable D8.1. The results of this activity have then been integrated into the first version of the SmartSociety platform.

2.1 Logical View

The logical view of the SmartSociety platform is presented in Fig. 1.

The core components which are part of the revised version of the SmartSociety are the following:

- **Application Runtime**: the application runtime is the component where the applications created by developers will run. The runtime provides all the necessary programming abstractions for interacting with the various internal components provided by the SmartSociety platform.

- **Orchestration**: the orchestration is in charge of the complete orchestration of a

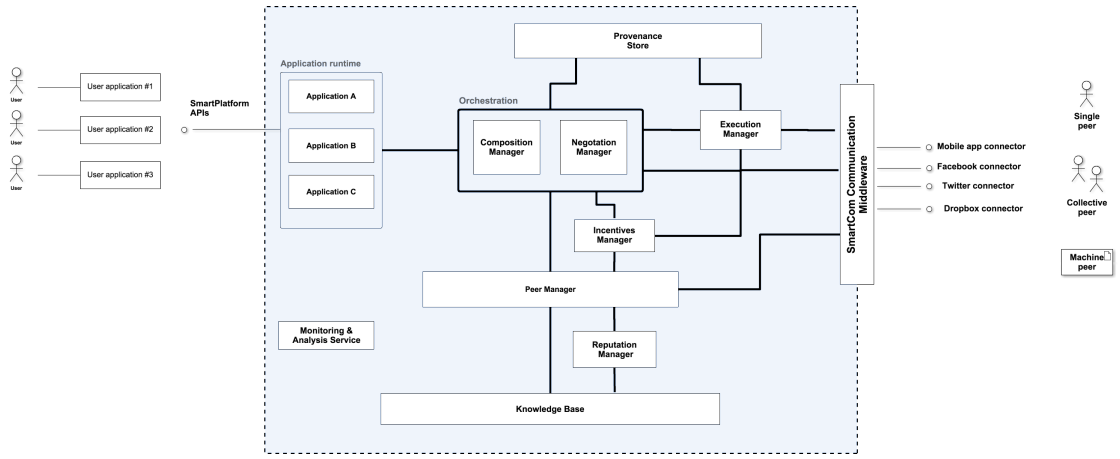

Figure 1: SmartSociety logical view.

given computation task. This includes the initial creation of a *composition* which can fulfill a given computation task, whereas a composition determines (i) the peers which can potentially execute such task, and (ii) the necessary interactions and/or external services which are needed to execute such task.

- **Peer Manager**: the Peer Manager is the module responsible for managing all peers-related information. This includes their profile, as well as any other information which will then be used by the Orchestration component for identifying possible peers which can execute a certain task. Besides individual peers, the PeerManager also manages collectives, which are groups of peers characterized by specific properties (see Deliverable XXX for additional details on collectives).

- **Reputation Manager**: the Reputation Manager handles the reputation of peers. Reputation is based on various metrics that the system will be able to compute, starting from peers' execution of tasks. A reputation score will be computed for every single peer, and this information will be available to the Peer Manager when

selecting peers for the execution of tasks.

- **Knowledge Base**: the Knowledge Base (KB) is shared among the various components of the platform, and contains an agreed ontology about peers, task, workflows, incentives, etc.. It provides the domain terminology, as well as all the semantic relations between terms. In particular, the knowledge base is responsible for ensuring that all component share a common understanding of the inputs/outputs between any two parties of the platform. The overall knowledge base will be divided into domains, which allow to capture the diversity of the entities being part of SmartSociety. A semantic reasoner will allow to perform semantic queries over the knowledge base.

- **Provenance Store**: the provenance store is the component responsible for keeping track of how the overall compositions are being created, executed and how the data managed by the platform is being transformed. It will play a fundamental role in the evaluation of peers reputation, as well as in ensuring that the system as a whole enforces the proper privacy constraints imposed by the peers.

- **Execution Manager**: the execution manager deals with the different interaction patterns that might be required to interact with peers.

2.2 Deployment View and Network Diagrams

2.3 Dynamic View

Fig. 2 provide the dynamic view of how the various components of the SmartSociety platform interact with each other, when running an application based on SmartSociety.

The sequence diagram can be summarized in the following steps:

- platform invocation: the starting point is a user application (e.g., a mobile application) which is exploiting the SmartSociety platform for running a human computation task (refer to Sec. 2.4 for a more concrete example). The request is directed towards an application deployed in the SmartSociety application and providing the necessary support for executing the requested task. The invocation will contain all the necessary metadata for fully describing the task, according to the APIs offered by the application running in the platform.

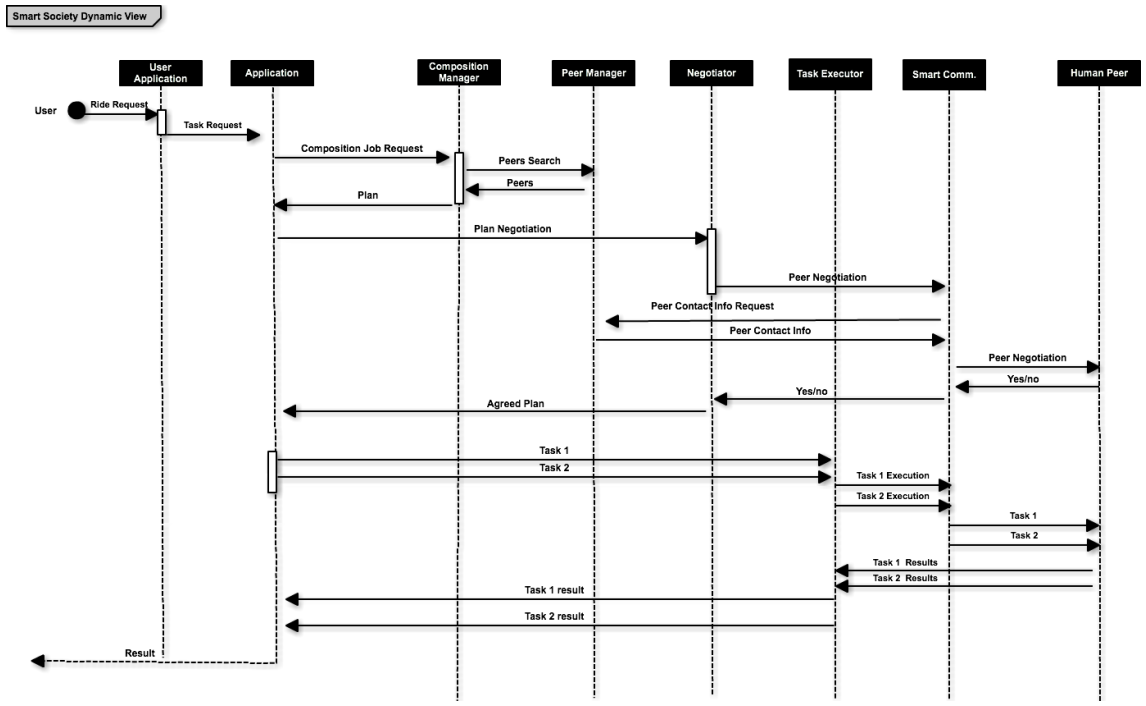

Figure 2: SmartSociety dynamic view.

- composition request: the application, after receiving the request from the user application, converts it into a *composition job request*. A composition is the request to create a composition of peers/collectives capable of executing the task. The composition job request happens through the programming abstractions provided by the runtime, which allow to fully characterize the task that needs to be executed.
-

2.4 Example: Ask SmartSociety

Ask SmartSociety is a simple Questions and Answer service enabled by the SmartSociety platform which has been used as the benchamark for implementing the initial version of the SmartSociety platform. It focus on tourism, which is the reference domain to be used for validating the SmartSociety vision.

Ask SmartSociety! will be a service where users can post questions in natural languages and peers can provide answers. Peers providing answers can be humans (individuals or

collectives) as well as machines (intelligent software agents). Peers can compose (forming collectives, hybrid or not) to provide answers. Answers can be ranked based on the reputation of the peers or on community ranking (similarly to stackoverflow). In some instances the user issuing the question can select an answer and provide feedback on the peer providing it. Two examples (grounded in the tourism application scenarios) can help in understanding the features of the Ask SmartSociety! service:

Next week Peter will fly to Venice. He will be busy in meetings during the day but wants to explore some hidden places at night. He could well explore various online tourist sites but he prefers to ask experts and local people. He could also google for relevant content, but he does not actually need an answer right away, he just needs to get it in one week. And having one system which allows him to query local experts, web-based recommendation services and incoming tourism institutions looks definitely appealing to him! Alice is visiting Milano during the next week. It is his first time in Milan, and she is looking for a restaurant in the city centre. Since it is spring time, she would love eat outside and therefore find a restaurant with a garden. Alice is also celiac, and she needs to find restaurants, which do have gluten free menus. She relies on the Ask SmartSociety! application for retrieving some suggestions on where to have dinner during her stay in Milan. She is looking for unconventional recommendations.

3 Interfaces Specifications

In this section we will list the components integrated at the moment and the API subset used.

3.1 Peer Manager

The Peer Manager (PM) API specifications can be found at http://demos.disi.unitn.it:8080/smartsociety/.

The PM is used for:

fying some requirements : This request is made by the Orchestration Manager. Endpoint used:
`???`

Creating collective : This request is made by the Application/Application Runtime at the end of the negotiation. Endpoint used:
`/peers/collectivePeer` (POST)

4 Description of Artifacts

here goes description of sw

5 Discussion and Furhter Integration Steps