A System Model and Applications for Intelligent Campuses

Attila Adamkó∗ and Lajos Kollár†
Faculty of Informatics
University of Debrecen
Debrecen, Hungary
Email: ∗adamkoa@inf.unideb.hu, †kollarl@inf.unideb.hu

Abstract—Nowadays, when Future Internet and Internet of Things (IoT) research is an integral part of the rise of computing, we need to react to new challenges. According to prognoses of Gartner and ABI Research, by 2020 there will be nearly 30 billion devices connected to the Internet of Things wirelessly. Furthermore, a new computing concept has appeared which had a vision where computing is made to appear everywhere and anywhere. These technological advancements that have occurred during the past decade in various domains, including sensors, wireless communications, location positioning technologies, and the web, allow the efficient collection and processing of a wide range of data. These ideas are applied at our University Campus where we established an extensible data management architecture, on the top of which value-added services are provided for various people living or working on the Campus. Both the architecture and a couple of applications that make use of it will be discussed.

I. INTRODUCTION

In the past seven years a new program has started its evolution. Future Internet research aims at bridging the gap between both academic and industrial community’s visionary research and large-scale experimentation. The key issues related to the future of the Internet go far beyond the technological dimension. We need to find answers (solutions) for new challenges. One of them is the Internet of Things (IoT) which highlights the opportunities lying in the sensors connected through wireless connections. They serve as an endless source of environmental data. Based on Gartner’s prediction1, it seems that the IoT market (without PCs, tablets, or smartphones) will grow to 26 billion units installed in 2020. This huge amount of data does not only need to be stored but efficient processing and (hopefully) real-time analytics are needed.

The next challenge is highlighted by the term of ubiquitous computing (ubicomp). This term refers to the computing concept where computing is performed in a transparent way: it appears anywhere and everywhere. It can be imagined as a “post-desktop model of human-computer interaction in which information processing has been thoroughly integrated into everyday objects and activities” [1], [2]. Recently, the major technologies that support distributed processing including the Internet, middleware and operating system products, sensors, etc. with all the related protocols are developed in a direction to support ubiquitous computing. This results in that ubiquitous computing can almost occur at any location using any device in any location, and in any format, and it is pretty sure that this trend will continue in the future. Borders between various computing devices including desktop computers, notebooks, tablets and smartphones are blurred, and even some of the devices that people often use evolve into computing (smart) devices. The recent rise of smart watches and Google Glass are evidences of this trend. By using such devices, users tend to interact with computers even when they are not aware of using them. Using the state of the art communication technologies and protocols, such a system can distribute processing seamlessly across a range of devices.

However, the idea of ubicomp cannot only be applied for distributed computing. A ubiquitous learning model within a pervasive smart campus environment is introduced by [3]. The main idea of the authors is to create learner profiles based on the learners’ behavior that can serve as a basis a ubiquitous learning environment that is context-aware and customized.

The third challenge is based on the power of the crowd. Crowdsourcing can be described as a special kind of outsourcing where the “contractor” of a business process is a large group of people. A University Campus is a good prospect of applying crowdsourcing as there are lots of people (students and staff) with different interests. The word crowdsourcing here is not used in a restrictive manner: the crowd itself can not only be the processor of information but it can be seen as content generator, as well. Many users share their ideas and provide either textual or multimedia contents using social media sites, they organize meetings and form groups, lots of usage data are generated while various applications are used, they travel to, from and within the Campus during which positioning and routing information might be provided and produced, etc. From the Smart Campus perspective, one of the main challenges is to collect content from various sources, the majority of which might be created at a later time. That is why we need a highly extensible system that is designed for change: a mechanism for defining emerging data sources and business processes should be created. Such an extensible architecture has been created that allows the extension of the system with new elements on both the data producer and consumer side. This is where the crowd could help us by adding new sources and new services. This approach is described in more details in Section IV.

1http://www.gartner.com/newsroom/id/2636073
Our primary goal was to establish an Intelligent Campus which is based on gathering data from the crowd, analyze them and provide feedback as value-added services that make users glad and in the meantime contributory as they would like to return. These services could be event-based, location-based or simply request-based, and will be discussed in more details later. However, for new data sources to be integrated, users can also develop their own information parsers which provides the extensibility of the system.

Naturally, data may arrive from several sources. The first direction is based on IoT. The collected data from the built-in sensors of smartphones, tablets, phablets, and embedded devices let the applications based on participatory sensing which become more and more popular and useful. However, this is only one direction, in a real-life environment we need additional (user- or application-generated) data sources in order to provide valuable services. These sources could include the local Education System, the Library Information System, several local websites, public websites, social sites and professional sites as well.

All of these new and emerging fields resulted in designing an architecture that is suitable to capture these interesting directions in our closer context at the University and serves as a basis for the development of applications that make use of the collected data. This paper highlights our proposal for a system model and organized as follows. Section II describes the context of our research and explains the reasons why we selected Smart Campus for our focus. Key challenges and goals are described in Section III. The architecture is discussed in Section IV while few sample applications built on top of it are introduced in Section V. Finally, Section VI contains some concluding remarks.

II. AN INTELLIGENT CAMPUS AS A SMALL SMART CITY

The FIRST (Future Internet Research, Services and Technology) project at the University of Debrecen targeted many of the challenges described in Introduction. Basic theoretical questions, examinations of network models, reassessment of network architecture, content management, and the creation of an intelligent application platform are also in the scope of the project.

The aims of one of its subprojects, Future Internet for Smart City Applications, are threefold:
1) to establish an open, mobile crowdsensing application platform for smart city applications,
2) to provide data management and knowledge discovery solutions for them, and
3) to develop prototype applications based on the common platform combining the data management solutions with real-time analytics.

Considering that there are lots of similarities between the challenges of a Smart City and a Smart Campus (term “Smart” might be used interchangeably with the term of “Intelligent” because they both intended to describe the capability of making adjustments that resemble human decisions, especially in response to changing circumstances), we chose to deal with campuses. A Campus is a good prospect of applying the aforementioned participatory sensing combined with other data and event sources and due to the number of people studying, working or even living at a Campus and considering their commitment of using novel technologies and applications, it is an ideal choice for developing value-added services based on crowdsensing. A cloud-based idea can be seen in [4] where education information has been put into focus.

The idea of Smart Campuses are not new. Many of the initiatives (see http://greensmartcampus.eu/ or http://smartgridcenter.tamu.edu/seci/ as examples) think primarily of Smart Campuses from an energy efficiency point of view. This approach deals with lots of environmental sensors that gather data which is then analyzed in order to optimize heating, consumption of electricity and water, etc. Of course, this is an important viewpoint but it is not related to the life of a Campus since any set of buildings might have similar challenges.

Smart Campuses are often approached from educational perspective. There are a number of challenges to be addressed: integration, assessment and learning, identity and ownership, security and privacy are all important aspects that needs to be dealt with [5]. The authors of [5] propose mobile cloud education as a ubiquitous, cloud-based way of providing anytime—anywhere context-aware learning using portable devices.

The iCampus initiative [6] as a generic framework not only identifies six base pillars (iLearning, iSocial, iManagement, iGreen, iGovernance and iHealth) that cover all the fundamental aspects that might rise up in a Campus environment but also defines a roadmap. This roadmap consists of four key development phases:
1) core technological and network infrastructure,
2) basic applications and services,
3) value-added applications and services (that are built on top of the basic services), and
4) premium applications and services.

Cloud computing and IoT are also identified as key ingredients of a Smart Campus in [4]. It demands the integration of the systems within the Campus ecosystem including teaching management and library management systems, financial systems and so on.

Our goal was to build some basic services first on top of which some value-added services can be added using an extensible framework that allows the crowd (first targeted only at Computer Science students) to create new applications. Based on the collected data about the usage of applications and services, some patterns regarding the operation of the community can be inferred. This information, after applying appropriate analytics, serves as a feedback to the services.

As a core technological infrastructural element, we have found that a publish-subscribe approach could be very useful and could fit into our vision. We choose an XMPP-based pubsub solution with crowdsourcing characterized by the following properties:

- Lots of potential users (Consumers)
  University of Debrecen has as much students and employees as the number of inhabitants of a medium-sized
Hungarian city. These users are consuming the provided services.

- **High variety of data sources (Producers)**
  Information in various formats coming from multiple sources should be integrated in both cases (e.g., timetable, academic calendar, information on consultation dates and times in case of Smart Campus, or energy consumption, traffic info, etc. for Smart Cities). Social media sites and geolocation sensors are examples of sources that are common in both domains.

- **Need for value-added services (Service Providers)**
  Service Providers give added value to the crowd-collected raw data. Basically we can say that this is why someone would like to use the applications (e.g., for being notified about a room change or finding a jam-free route between two points).

Examples of such value-added services include but not limited to:

- get notified when one of my favorite bands (based on my listening history from last.fm or Google Play Music) will play in a concert venue of the Campus,
- get notified when an event’s (class, concert, etc.) date and time is modified,
- based on the location data provided by some of my friends I can realize that they are going to have a lunch so if I am in a hurry I can join them,
- when preparing for a consultation with an instructor, get notified if too many students plan to visit the instructor’s office hour at the same time, and assist of reorganizing my schedule based on my task list.

### III. Objectives and Challenges

As a primary research objective, the development of an architecture which could fulfill the required data management and knowledge discovery demands mentioned above was needed. This architecture should address the challenges described in the following section and be as generic as it can be in order to integrate a high variety of data sources regardless of how the data is processed.

Another important objective was extensibility: the designed architecture should allow to add new sources and define their processing in an easy way. All of these needs led us to the publish—subscribe model and its XMPP implementation where extensibility is a primary concern [7]. It makes possible to include heterogeneous data sources into our system in a manageable easy way. Our domain, the Campus has lots of various (and most importantly, heterogeneous) data sources including the following which highlight the challenges:

- an Education Administration System called Neptun that contains information on course enrollments, timetable information of courses, exam dates and times, etc.,
- faculty members offering office hours, consultations, etc.,
- Education Offices of the various faculties offering office hours,
- Student Governments organizing events for students,
- the menus of the canteens located at the Campus,
- geolocation (e.g., GPS), WiFi or some other sensor data collected by smartphones or similar devices,
- data gathered by environmental and building sensors (temperature, humidity, air pressure, air pollution, etc.),
- a Library Information System that is able to tell whether a given book is available or not,
- social media sites (like Facebook or Google+) containing information on friends and ranges of interests of a person,
- professional sites (like LinkedIn) holding data on work experience and professional achievements (however, this is not necessarily the most important data source from Smart Campus perspective),
- bibliographic databases (like Google Scholar, DBLP or Scopus) that provide information of published journal articles or conference papers of researchers,
- event hosts of actually any events (like public lectures, concerts, exhibitions or whatever users might be interested in), and, which is essential,
- the crowd itself with the added value of the capability of generating content that is interesting for a set of people (or, to be more precise, consumers).

This list is far from being exhaustive, it is only an illustration to demonstrate how diversified sources of data can be used in a complex application. Valuable applications typically require integration of some data originating from several of these sources. Therefore, our main challenge was to create an architecture which allows the access of information from existing sources while also easing the addition of new sources in a seamless manner.

This addition task is highly depends on the source. While some of them could be gathered in an automated way (like sensor data), some of them may require manual interaction (e.g., canteens’ menus or instructors’ office hours); some of those data sources offer Application Programming Interfaces (APIs) to provide access to data (social media sites, for instance) while others do not have APIs so some web spiders are required to parse the data; some of the data sources provide built-in notification mechanisms (e.g., an event feed of a social network site) while others do not (for example, adding new office hours or changing the daily menu).

Another objective was the proper data management for the architecture. Considering the amount of data originating from those sources, we can state that a big data management solution is required. However, as we can foresee (and figure out based on our pilot applications described in Section V), the frequency of arriving data elements can significantly differ, some of them (e.g., inserting a new article into a bibliographic database or offer office hours for the forthcoming semester) might be quite rare while others (especially when collecting location data) can be very frequent. Our system model tries to break down this complexity by introducing new storage methods discussed in the following section.

Naturally, the development of value-added services may (and will) require data mining activities (including understanding the semantics of data in order to eliminate duplications),
as well, however, they are completely out of scope of this paper as we here focus only on the data management platform that can (and should) later be complemented with on-line data analytics solutions which are currently under development (by some of our colleagues). Data security is an important issue that needs to be addressed in the technology stack, however, our opinion is that it must be introduced at a higher level than the data management level, therefore we omit the discussions regarding that field.

IV. EXTENSIBLE DATA MANAGEMENT ARCHITECTURE

We needed an architectural framework that is designed to provide some kind of integration between software systems used at a Campus by collecting, organizing and analyzing data originating from various sources. We have chosen the Extensible Messaging and Presence Protocol (XMPP) as the underlying communication protocol [8], due to its extensibility and publish/subscribe model. XMPP defines a standardized way of event-based messaging which suited to our goals of notifying interested users about some events they subscribe to, instead of forcing service developers upon following a programming model that requires regular (and frequent) polling the state in order to check whether some events have occurred. Heterogeneity of data and diverse frequencies of events also justify this decision.

The concept of a Connector in our architecture (see Figure 1) describes a node which gathers data from given sources and sends updates to the Smart Campus infrastructure. The Smart Campus Collector Interface (shown in Figure 2) will receive a special XMPP message that is actually a notification that is used to determine whether updates based on the collected data should be approved or declined. The verdict will be _decline_ if a Connector sends a redundant piece of information. In this case, the repository will not be updated, of course. This might happen when, for example, the Research Connector collect the information about a publication that has earlier been found (e.g., the same publication is found in Scopus after Google Scholar) therefore adding it to the repository would lead to unnecessary redundancy.

Connectors here are compound entities without any deep business logic. Due to the extensibility of the framework, the crowd (by now only those members of the crowd who know how to program in a programming language) can develop and add their own connectors. Based on the data source to be processed it can either call the API of a data source to retrieve the required information or they can encapsulate a web crawler or a parser that can process its data source and extract the information that is provided by the connector itself. The only requirement that should be fulfilled when preparing a Connector is the ability of handling the proper XMPP message structure.

This makes our architecture very powerful as it does not limit the possible data sources and also allows the collection of some specific information. For example, if a couple of students prefer to have a menu lunch at the small restaurant near the Campus they can develop a connector that parses the restaurant web page to provide information on the daily menu. This emphasises the role of the crowd: if connectors would only be developed by the university staff, this menu will probably never become available as (presumably) only canteens located at the Campus would be taken into consideration. However, this cheap restaurant might become an important part of the life of Campus inhabitants.

The layers of our Smart Campus architecture are shown in Figure 2. Data sources are placed in the undermost layer since their presence is fundamental for the other parts of the framework.

The next layer is dealing with Data Access. This layer has a very important role in the architecture since it is the layer where the retrieved data are collected and made available to the Central Intelligence layer.

The Smart Campus Central Intelligence (SCCI) component in our architecture provides an interface between the information sources including both the incoming events (XMPP server) and the information stored in the database and the Web services layer (Figure 2). It provides a couple of unified data models related to some of the most notable domains in a life of a Campus: educational data, research data, social data (with friend of and classmate of relationships), etc. SCCI layer also allows to define and validate business rules and this is where the Analytics module resides. Now we have only a couple of simple analysis that does not exceed the scale of a medium-level database query but deeper analytical capabilities are planned to be added as this is how we can provide better value-added services.

On top of the SCCI, web services are defined. We decided
to provide our services via web services since various front-ends need to be supported. Web applications and different mobile platforms (Android, iOS, Windows Phone) are natural development targets (see uppermost layer) but due to the application of standard web services technologies, high level composition of those services is also possible.

Figure 3 shows a service-oriented view of the architecture. The bottom layer, named Data Collector, is the one where Connectors reside. These connectors are responsible to handle data sources and retrieve data from them.

The Data Management layer contains databases that are populated with data gathered by connectors of the underlying layer. As of today, due to the characteristics of the data we have both a relational and a graph database as a back-end. Those pieces of information that are well-structured (e.g., course and timetable information) are stored in a normalized relational database. However, for semi-structured or unstructured information the rigid structure of relational tables are not appropriate. We have selected neo4j as a solution but as now we have only a limited number of users that will hopefully significantly increase in the future, this field needs further investigation. Fortunately, graph databases scale well so probably it will be enough to add a couple of new nodes and we will not need to change the underlying database.

However, as databases grow and the more data are available about a group of entities, the more possible is that we have to sacrifice uniformity and structure for the sake of wider knowledge. The result is a highly semi-structured system. It is not just collecting large volume and variation of date but also understanding the relationship between entities by mapping their connection into our system. With this method we established a data store which can be easily extended. When a new—and previously non-existent—source is attached to the system the only task we need to perform is to add the new nodes and proper edges to the database.

If we examine the kinds of data in a Campus we can find that there are lots of more or less static data like personal data or course data that are rarely changed while another class of data are “fresh” (real-time or almost real-time) like sensor data or information on various events that become deprecated as time progresses. It is observable that while it is important that frequently changing data (like sensor data, news feed) should easily and quickly be accessed when it is fresh, after a while they become outdated (but analytics might need it). This is the same idea that are applied in traditional data warehouses: “old” data from OLTP systems are loaded into OLAP systems. We applied this approach when separated current data from historical data. So we introduced a separate HistoricalDB part which is populated after a defined period of time. This separation can speed up the queries targeted for “fresh” data because the data store cannot grow to a huge one slowing down all the searches. However, the other part, the historical one introduces new challenges as well. At one
hand, efficient handling of big data. On the other hand, proper support for the Analytics module to find new and relevant information from that huge amount of data. In fact, this is something that probably needs to be reconsidered after finding the most appropriate database solution.

Applications that are created based on Web services are at the top of the view. Some applications that have been developed and deployed so far will be detailed in the next section.

V. APPLICATIONS

Based on the proposed architecture, we created few sample applications. Most notably, a calendar application has been implemented which integrates calendar events from various sources including the timetable, office hours, schedule of public lectures or events held at the faculty. Currently there are two applications which consume these services: one of them is a Web application implemented in Java while the other one is native Android application. The latter is location-aware so based on WiFi access points it figures out where the user is (which corridor of the building) and shows only those relevant pieces of information that are close to the user. Users can subscribe to the events they are interested in and they can also rate the events. The gathered information can be analyzed in order to provide such value-added services like recommending and reorganizing (e.g., when an office hour is fully booked) events. The app’s built-in XMPP-client allows receiving a message when an event is cancelled or re-scheduled which shows the power of the architecture based on the publish–subscribe approach.

Another Android-based application for gathering information coming from the various sensors (including GPS data) of a smartphone has also been developed. These data can be analyzed in order to find frequent routes (trajectories) of the user, on the top of which value-added services like avoiding over-crowded places or routes or organizing a meeting with a friend who also walks the same route. A third application that is currently under development aims at a crowd-assisted learning support for students who learn how to program. This is a web application with a JavaServer Faces (JSF) front-end used for asking help on programming exercises. The members of the crowd can give feedback on a student’s upload as a rating from one to five stars with an optional comment. Feedbacks can also be rated based on their usefulness. In order to avoid cheating, feedbacks and ratings are anonymized. Based on the data gathered via feedbacks and ratings, those crowd members who give more valuable responses can be identified which improves the overall confidence of the system.

VI. CONCLUSIONS AND FUTURE WORK

The sample services are now in a beta-testing phase and they will be released soon at the Faculty of Informatics, University of Debrecen. The applications justify our architecture, however, some fine-tuning might be necessary. The architecture fits well into the more general publish/subscribe based architecture of Smart City and Smart Campus applications. The developed architecture is extensible with new data sources (appropriate Connectors need to be developed when adding a new source) providing the capability of integration of heterogeneous data. In the future, development of the Analytics module is a major goal since providing good analysis of the collected data can add more value to the services.

ACKNOWLEDGMENT

The publication was supported by the TÁMOP-4.2.2.C-111/KONV-2012-0001 project. The project has been supported by the European Union, co-financed by the European Social Fund.

REFERENCES