ABSTRACT
There are different types of information system architectures, but there has not been a structural comparison between the different types. I develop in this paper a structured approach to assess the three main Information System architectures, Enterprise Resource Planning, Service-Oriented Architecture and Multi Agent Systems, based on a structured approach. Furthermore, I provide in this paper an extensive analysis of the key characteristics of the three main IS architectures. In this paper, I develop a systematic and structured approach to pairwise compare and assess these key characteristics of the main architectures based on the IS development framework as developed by Zachman. The aim of this paper is twofold, first, to develop a framework which is suitable for comparing IS architectures and second, comparing the three architectures.

The framework I developed is based on the Zachman framework and the views of the original framework are grouped together in due to the complexity of comparing the original framework for the three architectures, this resulted in a new 6x1 framework. The results of the comparison show that there are several key difference between the architectures in the areas of what kind of data is stored, the ability to keep working with legacy systems, the amount of complexity and uncertainty in which these architectures can operate and how the network which corresponds with these architectures is arranged.

Keywords
Enterprise Resource Planning (ERP), Service-Oriented Architecture (SOA), Multi Agent Systems (MAS), Information Systems Architecture (ISA), framework, characteristics.

1. INTRODUCTION
There are different types of information systems available; in order to choose the one that best fits the needs of an organization, first a comparison between the different types of systems based on characteristics has to be made. Such a comparison does not exist and that is the reason for this paper. There are two goals in this paper, first to develop a framework which is able to compare IS architectures and second to compare the actual IS architectures. In this paper the underlying architecture of information systems will be compared, more specific three types of architectures, namely Enterprise Resource Planning (ERP), Service-Oriented Architecture (SOA) and Multi Agent Systems (MAS). In order to do so, Zachman’s Information Systems Architecture (ISA) framework is used as a basis for a framework to compare the architectures. Further in this paper the problem is stated, research questions are proposed, the method of research is explained, a framework for comparison is offered, the characteristics and workings of the architectures are described in order to compare the architectures and a framework is applied and conclusions are given.

2. PROBLEM STATEMENT & RESEARCH QUESTIONS
In this section the main problem statement is presented. Several research questions are identified to specifically touch upon the key problems.

2.1 Problem Statement
There has been a lot of research conducted about what different types of information system architectures there are available, what there specifics are, how they are implemented, how to select vendors and what the key success factors are. However, there is no clear comparison in this broad field of architectures based on the differences of these architectures. In this paper three kinds of architectures are compared using a structured method.

2.2 Research Questions
The main research question of this paper is:

What are the differences of ERP, SOA and MAS architectures and how can these architectures be compared?

In order to answer this question completely, three sub questions are proposed. These sub questions help to provide partial answers in order to answer the main question.

- Which framework can be used to compare the three architectures?
- What are the characteristics of each of the three architectures?
- What are the differences between the three architectures?

3. METHOD OF RESEARCH
To get the answers to the research questions above, a literature study is conducted. The initial method for getting the appropriate literature was to search in the top 10 Information System journals, listed by Levy and Ellis[8]. This was done by entering the following keywords on ISI Web of Knowledge; “Enterprise Resource Planning”, “Service-Oriented Architecture”, “Multi Agent Systems” and their respective abbreviations. As an addition to the query the top 10 journal titles were added in the publication name field. The first selection was made on title, if the title looked promising the abstract was read. When an abstract was considered useful to the subject, the paper was used as a resource for writing the paper. This yielded far too little useful results to use for this...
paper, so the search was widened. The same keywords and selection procedure were used to search the Scopus database for papers. Also Scopus has the option for searching in literature reviews, this was done as well. This yielded far more results and the results were sorted in descending order on number of citations. The results can be found in table 1 below.

Table 1: Results of literature search

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Papers selected on title</th>
<th>Papers selected on abstract</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERP</td>
<td>11</td>
<td>4</td>
</tr>
<tr>
<td>SOA</td>
<td>14</td>
<td>4</td>
</tr>
<tr>
<td>MAS</td>
<td>14</td>
<td>4</td>
</tr>
</tbody>
</table>

4. INFORMATION SYSTEMS ARCHITECTURE

The Information Systems Architecture framework was first developed by Zachman and later extended by Zachman and Sowa[13]. The framework is widely used to design, analyze, and classify enterprises. The goal of the framework is to describe the real world into concepts that represent an enterprise and how everything fits together. The framework is a matrix of six columns and six rows. The columns represent the real world, or abstractions. They consist of data, function, network, people, time, and motivation. The rows represent different views or perspectives of the information system; they are scope, enterprise model, system model, technology model, components, and functioning system. Each cell of the matrix has its own point of view on the information systems. In a single cell of the framework there is a unique representation of the architecture. This representation can either be described graphical or textual. The advantages of this framework are that it can structure and describe complex systems. A segmented and portioned view of the individual variables of the information systems is presented, thus making it possible to compare architectures. The original ISA framework is shown in table 2.

Table 2: Original ISA framework

<table>
<thead>
<tr>
<th></th>
<th>Data (what)</th>
<th>Function (how)</th>
<th>Network (where)</th>
<th>People (who)</th>
<th>Time (when)</th>
<th>Motivation (why)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope (planner)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enterprise model (owner)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System model (designer)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Technology model (builder)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Components (sub-contractor)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Functioning system</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In order to use the framework to compare the three different architectures a change has to be made to the framework. All the views, rows, will be considered as one. This is done, because every cell in the original ISA framework has to be filled in completely in order to keep the normalization of the framework[1]. Also, in order to fill every cell, detailed information about the implementation of an architecture is necessary because there are six different views of the architecture. Furthermore a comparison between the 36 cells of each of the three frameworks is not feasible due to the complexity of comparing the three filled out frameworks. The column People, or who, can be read in this case as software applications or agents. The framework is applied in paragraph 6.

5. CHARACTERISTICS OF THE ARCHITECTURES

The three architectures that will be described and compared are ERP, SOA and MAS. The characteristics that are described are characteristics of the architectures themselves, how they work and communicate. Characteristics about implementation, impact to an organization, management implications and user implications are not described.

5.1 Enterprise Resource Planning

ERP is a packaged software solution with the aim to integrate all of the business processes and functions into a single view of the business. To get this single view of the business, ERP systems use a single underlying database to store data. This database is used to support the business’ core processes and administration. ERP intends to support all the business processes, including production, inventory management, sales, planning and money related processes.[6] ERP systems consist of different modules, each module contains different functions and can be bolted onto the existing ERP system or installed on its own. Examples of modules are customer relationship management, supply chain management and business intelligence. With the development of these modules, the fields in which ERP is employable shifts from not only the original production/manufacturing environments, but other environments as well[9] The differentiating feature of ERP solutions compared to other single database solutions, is the fact that ERP is highly customizable. Businesses differ in the type and amount of processes and functions, thus making it key for a (generic) ERP package to be highly customizable or easy to individualize in order to meet the specific needs of a business. Since ERP is an all-in-one solution, a characteristic is that the graphical user interface is the same throughout the different parts of the application. Other characteristics are that ERP systems are able to handle large volumes of transactions and that is open in terms of the choice in hardware and software platforms. The complexity of ERP systems is considered to be a drawback due to the integration of all the business processes in one solution.[6]

ERP can be divided in two categories, concept oriented ERP and system oriented ERP. Concept oriented is the operational and strategic use of ERP and system oriented is the implementation and maintenance side. The following definition can be used to illustrate this. “ Seamless integration of processes across functional areas with improved workflow. Standardization of various business practices, improved order management, accurate accounting of inventory and better supply chain management. ERP systems on the other hand are merely the vehicles through which this is accomplished.”[5] Key terms of the concept of ERP are integration, standardization, extension and assurance of future flexibility. ERP implementations tend to have high centralization and low flexibility. The fundamental benefits of ERP systems are efficient transaction processing and storing data for processing in a structured, organized and centralized way. All the data of an ERP system is stored in a single integrated database.
The ERP concept is based upon a transaction processing systems like Manufacturing Resource Planning, these systems were designed to link processes and forms together in order to help planners with their work. ERP was designed to integrate the smaller existing systems of a company, to maintain real-time accountability of resources within the whole firm. Subsequently this led to elimination of conflicting information, reduction in data redundancy, standardization of interfaces and system wide access and security settings. Nowadays it is important to ERP systems to be able to add additional modules on top of the existing system.

There’s only a limited amount of research in the ERP area, due to fact that many researchers believed that contributions to the ERP research field had to come from programmers and human-computer interaction specialist. Another reason why ERP research is behind due to the crippling factor that “getting the system to run” often dominates discussions within companies. The sheer complexity of most ERP systems makes it hard to link benefits with ERP investments. Research has been done mostly about cost, time, success, vendor selection and package implementation and not on the fundamentals or characteristics of ERP.[4]

5.2 Service-Oriented Architecture

When applications have to interoperate with each other, most of the time problems arise with for example different data exchange formats and connectivity among the applications. In order for all the different applications to work together, you need “enterprise policies which must readily authenticate and authorize the parties involved in different interactions. A SOA, by requiring that policies be made explicit, can organizationally enforce compliance with these policies, thus simplifying the system’s management.”[6]

SOAs address “fundamental challenges of open systems, which are to operate efficiently and achieve coherence in the face of component autonomy and heterogeneity.”[3]

SOA was developed to meet the requirements of loosely coupled, standards-based and protocol-independent distributed computing. This is done by using services, “which are well defined, self-contained modules that provide standard business functionality and are independent of the state or context of other services.” The SOA modularizes large applications into services. Services have three essential properties, namely;

- Self-containment, the service maintains its own state
- Platform independence, the interface contract of service only have platform independent assertions
- Services can be dynamically located, invoked and recombined

This enables a SOA to be independent of any specific technology. Also it helps to overcome problems in the areas of application integration, transaction management, security policies and the means to keep working with legacy systems. The goal behind developing SOA was to come up with an architecture that supported application integration and to have this integration run seamlessly. A focus of SOA is allowing companies to develop, interconnect and maintain enterprise applications and services efficiently and cost-effectively by creating a design style, technology and process framework. Interfaces are the means of accessing service implementations, not by directly accessing the connected applications itself.

Services in a SOA have the following main characteristics;

- All functions are defined as services.
- All services are autonomous, this means that external components do not know how services perform their functions, they only get the results and not the internal workings.

- Interfaces are invocable, this means when a service is called upon via an interface, it is irrelevant whether the service is local or remote, how the service internally works nor infrastructure components are required.

A typical communication in a SOA consists of two parties, a service requestor and a service provider (client and server). They communicate through service requests. Many of today’s SOA implementations are not that elaborate, this is due to the fact that either the SOA provides limited functions or that security arrangements are not sufficiently in place.[10]

Companies that implement a SOA are able to reduce integration of projects and maintenance costs by at least 30%. [2] A SOA has the ability to keep using legacy systems, but this comes with a drawback. A wrapper has to be coded on top of the legacy system, instead of invoking the system directly. This has a negative impact on performance, but writing the code for these wrappers is much simpler then the code for the direct access. Another big advantage of using a SOA implementation is that the overall maintenance effort is reduced because of access rights to legacy systems are very restrict and changing functionality requires redevelopment of the system. This in contrary to wrappers which need much less effort to change them.[11]

5.3 Multi Agent Systems

MAS are systems where agents, autonomous software applications, collectively and individually work to meet their goals by interacting with each other. This architecture can be used as an IS when the different agents collaborate to create a single view of an environment. In a MAS it is important to have a clear understanding of what agents exactly are and what they do. Agents are autonomous software entities that can flexibly achieve their objectives by interacting, or bargaining with other agents in a MAS. They interact with each other in terms of high-level protocols and languages.[15] Since agent systems are developed with the idea that agents have to work together in order to meet certain goals, it is crucial that agents can efficiently interact with each other. “The most important issue in the architectural design of agent-based system is how agents interact with each other to achieve system goal”. [7] Agents are designed to fulfill a specific goal and in order to do so they have control over their own actions and behavior, thus being autonomous and can meet these goals in a flexible way. Agents may have to interact with other agents to solve problems or reach goals, interaction can happen in various ways, information interchanges, requests for particular actions and cooperation, coordination and negotiation in order to arrange interdependent activities. Agents interactions differentiate from other computational models in two ways, first these interactions are conceived in terms of which goals should be followed at what time and by whom. Second, agents have the ability to make “run-time decisions about the nature and scope of their interactions and to initiate (and respond to) interactions that were not foreseen at design time.”[5] Key specifics of such architectures and/or systems are decentralization, flexibility, dynamic, modularity and different stakeholders. This makes the architecture suitable for dynamic and unpredictable situations and environments. MAS can be seen as an organized society of individuals in which each agent plays specific roles and interacts with other agents according to protocols determined by the roles of the involved agents. Agents have the following characteristics:
There are two major drawbacks with using agent based systems, these drawbacks are associated with the fundamentals of agent systems itself. First the patterns and outcomes of interactions between agents are inherently unpredictable, because agents itself are autonomous, flexible and they have control over their own actions. The second drawback is that predicting the behavior of the whole systems based on its constituent parts is extremely difficult because of the strong possibility of emergent behavior. [5]

Agents interact (cooperate, coordinate or negotiate) with other agents in pursuit of common goals or own interest. [15]. There are two types of systems in which agents occur, Multi-agent software systems and open systems. In the first one all the different agents work towards a common goal or objective. To do so, they have to interact with another by means of cooperation and communication to meet a predetermined goal. In the second system agents are not necessarily designed to work together towards one goal, agents can have self-interested behavior. In an open system agents can join and leave the environment, making it more dynamic then a MAS, were all agents are known and can trust each other. An agent can have one or multiple roles, a role defines what an agent is supposed to do. [14]

6. RESULTS

### Table 3: ERP

<table>
<thead>
<tr>
<th>Data (what)</th>
<th>Function (how)</th>
<th>Network (where)</th>
<th>People (who)</th>
<th>Time (when)</th>
<th>Motivation (why)</th>
</tr>
</thead>
</table>

### Table 4: SOA

<table>
<thead>
<tr>
<th>Data (what)</th>
<th>Function (how)</th>
<th>Network (what)</th>
<th>People (who)</th>
<th>Time (when)</th>
<th>Motivation (why)</th>
</tr>
</thead>
</table>
7. DISCUSSION

As a result of filling in the framework with the three architectures, several similarities emerge. All the architectures are built in a (somewhat) modular form. ERP and SOA can extend their functionality by adding modules to the existing architecture, while agents are modular by design. All the three architectures have similarities in defining communication within the systems. MAS and SOA define the protocols or interface more explicitly because it is key for these architectures that the different parts can interoperate with each other. ERP is more implicit since all the communication is within the ERP package.

There are a number of differences between the three architectures as they emerge from the applied. ERP and MAS keep data of the enterprise and/or environment in order to (re)act with this information, or process it. With ERP it is crucial to get all the relevant data from an enterprise into the package, because the main function of ERP is to process this data and create a single view of the enterprise. MAS collect data from the enterprise or environment to weigh them in order to act to reach the pre defined goals. SOA on the other hand, is only concerned with data communication, and does not process this data. ERP and SOA have the ability to use existing or legacy systems, the first one integrates them in one package, the second one the services and tools to let them communicate and interoperate with each other. MAS does not do neither, or the existing systems need to be completely overhauled. MAS are able to cope with flexible and dynamic environments, not all the components or other agents have to be known, or in the case of a complex problem not everything has to be mapped and coded, in order to come up with a working system. This should make MAS better for working in dynamic and rapidly changing environment or if goals often change. For ERP everything has to be identified to create the all-in-one solution that ERP is, in order to get the system working. This creates the ability for streamlining business processes and making them more efficient. This comes from the evolutionary path that ERP has gone through and started with manufacturing resource planning. SOA provides a bit of both, applications can be added and removed without reengineering the SOA itself. It provides a interoperability platform that is independent whether the whole enterprise or environment is using the SOA or not. This difference also comes up when looking at the network. ERP has a network that is closed, it is known to the ERP package where all the parts of the system are. In MAS agents can freely enter and leave the environment, thus making the network itself open to change. SOA isn’t concerned whether a services is locally or remotely invoked, as long as the requestor uses the right interfaces and protocols. All the architectures can have the ability to be real-time representations of the real world, but in case of SOA it depends how the underlying applications are designed and in MAS it depends on the agents itself.

The frame is used for comparing these three architectures should allow other architectures to be compared in the same manner. As all the views are grouped together, this makes for a more high level view of an architecture, other views can be added or changed in order to come up with a more elaborate or detailed view of an architecture. The cells of the framework are filled in textual, but this can be altered to represent architectures in a graphical way.

8. CONCLUSIONS AND FUTURE WORK

The framework used in order to compare the three architectures is a framework which is based on Zachman’s ISA framework. All the views are combined due to the complexity of comparing the 6x6 framework times three architectures. The new framework is a 6x1 framework, were only the abstractions, or columns, of the original ISA framework are used. There are several differences between the three architectures as the framework is applied to the architectures. ERP and MAS keep data of the enterprise and/or environment in order to (re)act with this information, or process it. SOA on the other hand, is only concerned with data communication, and does not process this data. ERP and SOA have the ability to use existing or legacy systems, the first one integrates them in one package, the second one the services and tools to let them communicate and interoperate with each other. MAS does not do neither.

There are several differences between the three architectures as the framework is applied to the architectures. ERP and MAS keep data of the enterprise and/or environment in order to (re)act with this information, or process it. SOA on the other hand, is only concerned with data communication, and does not process this data. ERP and SOA have the ability to use existing or legacy systems, the first one integrates them in one package, the second one the services and tools to let them communicate and interoperate with each other. MAS does not do neither.

MAS are able to cope with flexible and dynamic environments, not all the components or other agents have to be known, or in the case of a complex problem not everything has to be mapped and coded, in order to come up with a working system. For ERP everything has to be identified to create the all-in-one solution that ERP is, in order to get the system working. SOA provides a bit of both, applications can be added and removed without reengineering the SOA itself. It provides a interoperability platform that is independent whether the whole enterprise or environment is using the SOA or not. This difference also comes up when looking at the network. ERP has a network that is closed, it is known to the ERP package where all the parts of the system are. In MAS agents can freely enter and leave the environment, thus making the network itself open to change. SOA isn’t concerned whether a services is locally or remotely invoked, as long as the requestor uses the right interfaces and protocols. All the architectures can have the ability to be real-time representations of the real world, but in case of SOA it depends how the underlying applications are designed and in MAS it depends on the agents itself.

The frame is used for comparing these three architectures should allow other architectures to be compared in the same manner. As all the views are grouped together, this makes for a more high level view of an architecture, other views can be added or changed in order to come up with a more elaborate or detailed view of an architecture. The cells of the framework are filled in textual, but this can be altered to represent architectures in a graphical way.

### Table 5: MAS

<table>
<thead>
<tr>
<th>Data</th>
<th>Function</th>
<th>Network</th>
<th>People</th>
<th>Time</th>
<th>Motivation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(what)</td>
<td>(how)</td>
<td>(where)</td>
<td>(who)</td>
<td>(when)</td>
<td>(why)</td>
</tr>
<tr>
<td>MAS</td>
<td>-Specific goal(s) for agent(s) and/or its role(s)[5]</td>
<td>-Facilitate agent interaction[15]</td>
<td>-Agents can join and leave an environment, thus expanding and reducing the size of the environment and network[14]</td>
<td>-The different agents are autonomous, reactive, proactive, rational, adaptable and situated in an environment. Thus making agents unpredictable when they act.[5, 12, 15]</td>
<td>-Problem is too complex for a single software solution[12]</td>
</tr>
</tbody>
</table>

The different architectures have similarities in defining communication within the systems. MAS and SOA define the protocols or interface more explicitly because it is key for these architectures that the different parts can interoperate with each other. ERP is more implicit since all the communication is within the ERP package.
protocols. All the architectures can have the ability to be real-time representations of the real world, but in case of SOA it depends how the underlying applications are designed and in MAS it depends on the agents itself.

This research in limited due to the constrains as given in paragraph 6, this makes more research necessary to come up with a more complete comparison between the different architectures. Also the comparison is between three architectures, whilst there are more information systems architectures. The first step has been made to compare the three architectures based on characteristics, the next step will be that this comparison is used to map IS architectures to business structures.

9. REFERENCES