



## SolutionsWorks BeePlex 1.0

*January 2, 2008*

<mailto:tfigliulo@solutionsworks.com>

<mailto:ikemsley@solutionsworks.com>

**Confidentiality Statement:**

*This document contains confidential proprietary and trade secret information belonging to SolutionsWorks, Inc. and shall be treated as such. No Section of this document shall be copied or otherwise reproduced or distributed, in whole or in part, without the prior written consent of SolutionsWorks. All material contained in this document is protected by the copyright laws of the United States.*

**TABLE OF CONTENTS**

**1 ABSTRACT .....4**

**2 BEEPLEX INTEGRATION NETWORK .....4**

    2.1.1 Specifications .....5

**3 BEEPLEX TECHNOLOGY OVERVIEW .....6**

    3.1 BEEPLEX ARCHITECTURE .....8

    3.2 THE PRIMARY FEATURES OF BEEPLEX .....9

**4 BACKGROUND AND RELATED RESEARCH.....10**

**5 ABOUT SOLUTIONSWORKS .....10**

    5.1 LINKS .....11



## **1 Abstract**

BeePlex gives you access to the "Internet of Things" based on a state of the art implementation of the ZigBee and IEEE 802.15.4 protocol. The unique combination of ZigBee, IEEE 802.15.4 and BeePlex is ideal for the implementation of a wide variety of easy to manage, low cost, low power and reliable control and monitoring applications that will work anywhere and everywhere. This includes industrial and agricultural environments, no matter how daunting the challenge.

BeePlex EDA uses a "message bus" and XML messaging which allows you to implement rich work and process flows. Reliable store-and-forward messaging guarantees delivery of events and ensures that your system is robust. Because BeePlex provides you with an inherently decoupled architecture it gives you unparalleled flexibility and component reuse. Incremental development of your integration solution is greatly facilitated. BeePlex is quick and easy to deploy and allows for the versioning of components. Add ZigBee wireless technology to this and you have an enterprise ready wireless solution that is truly scalable with unparalleled adaptability to ever changing and challenging requirements. BeePlex features all the tools necessary to run a mission critical Service Oriented Architecture (SOA) including security, centralized management, built in monitoring and fail-over capabilities in addition to extensive statistical reporting.

Data collection is only one part of the equation. BeePlex provides a crucial complementary component to assimilating data – namely organizing masses of discrete data into a logical schema so that it can be used effectively. Data in the BeePlex system is strongly assigned into taxonomies according to type and arranged into a flexible system of abstract categories or "topics." This allows information to be ordered into useful classifications that can be filtered by receivers or users of the information according to geographical area, type of event, severity, or into a hierarchy that is meaningful to the recipients. This allows data to be logically channeled or streamed and it also allows nodes that emit information to be logically decoupled from receiving nodes, greatly adding to the flexibility and scalability of the overall system. This technology has been adapted from enterprise application messaging techniques and is usually referred to as the "publish/subscribe" or pub/sub model.

## **2 BeePlex Integration Network**

BeePlex scope of control spans from the small local network of devices up to integration with large scale enterprise.

BeePlex decouples ZigBee networks by implementing ZigBee controls as BeePlex devices called ZeeAgents.

ZeeAgents are autonomous sensors, monitors and control devices that perform a variety of tasks depending on the type of ZeeAgent device.

- Communicates using open-standard ZigBee RF
- Participates in a ZigBee wireless mesh-network
- Sends or consumes message packets using a documented XML schema containing a "topic" element

BeePlex is process-driven. The ZigBee specification's default network state is data driven, a attribute of most ZigBee networks. BeePlex operates using a network of distributed agents rather than point to point communications between ZigBee devices.

Sending data with a given topic name is called “publishing” data to a “topic”. Publishing data is a fire-and-forget activity from the device point of view and the routing and preservation of the message is the responsibility of the underlying BeePlex message transport system.

Data in a BeePlex is strictly arranged according to type and categorized using a logical hierarchy of abstract *topic names*. Topics are analogous to subject lines in e-mail messages but are far more powerful. Topics can be specified as a hierarchical structure like the directory path in a computer file system. It’s up to the customer to decide exactly what topic names to set for a device, but once set, the device will associate all data it sends with that topic. In essence the data is “labeled” so that any device or logical entity can receive messages with a particular type of label simply by registering its interest in a particular type of “label” with the BeePlex network.

Publishing data on a network is differs from point to point ZigBee solutions in key areas:

In a point to point ZigBee network the user is locked into a predefined, rigid network hierarchy. Special network expertise is necessary when the physical restraints necessitate network hierarchy changes.

BeePlex decouples ZigBee devices from the physical network to take full advantage of the underlying mesh network topology enabled by the ZigBee specification.

All messages are *broadcast* and so are logically available to all ZeeAgents and other BeePlex entities in a network. A BeePlex network can securely cover a large urban area. The network can bridge the public Internet or a private intranet. The network bridge enables automated notification using COTS notification product suites. No special and expensive custom network applications are needed for BeePlex network notification. The installation can use whatever notification they are accustomed to using. BeePlex can be fully integrated in to any enterprise level system.

BeePlex uses a component architecture. This enables components to be deployed in order to extend the BeePlex infrastructure. A component level extension to BeePlex is a device called a TiniAgent which connects the ZigBee RF mesh-network to an intranet or to the public Internet. TiniAgents are small, low-power devices that provide a gateway into the sophisticated BeePlex Event Driven Architecture (EDA). The BeePlex EDA consists of BeePlex applications that are hosted on one or more Windows or Unix servers called BeePlex *Agents*. ZeeAgents are virtual containers that contain the sensing application, automated actions, and easily modifiable rules-based programming to enable flexibility to respond to changing threat levels. Agents host virtual services, which can be configured using a centralized graphical console to perform tasks ranging from updating a database or sending e-mails or text messages.

Agents use the same publication and subscription (pub/sub) protocol as ZeeAgents. In essence they are virtual implementations of ZeeAgents but use their own TCP/IP messaging protocol to communicate rather than ZigBee and the IEEE 802.15.4 specification.

### **2.1.1 Specifications**

ZeeAgent – Semi-autonomous container objects that are part of the embedded software system hosted by the sensor node. This is part of the application layer in the OSI model.

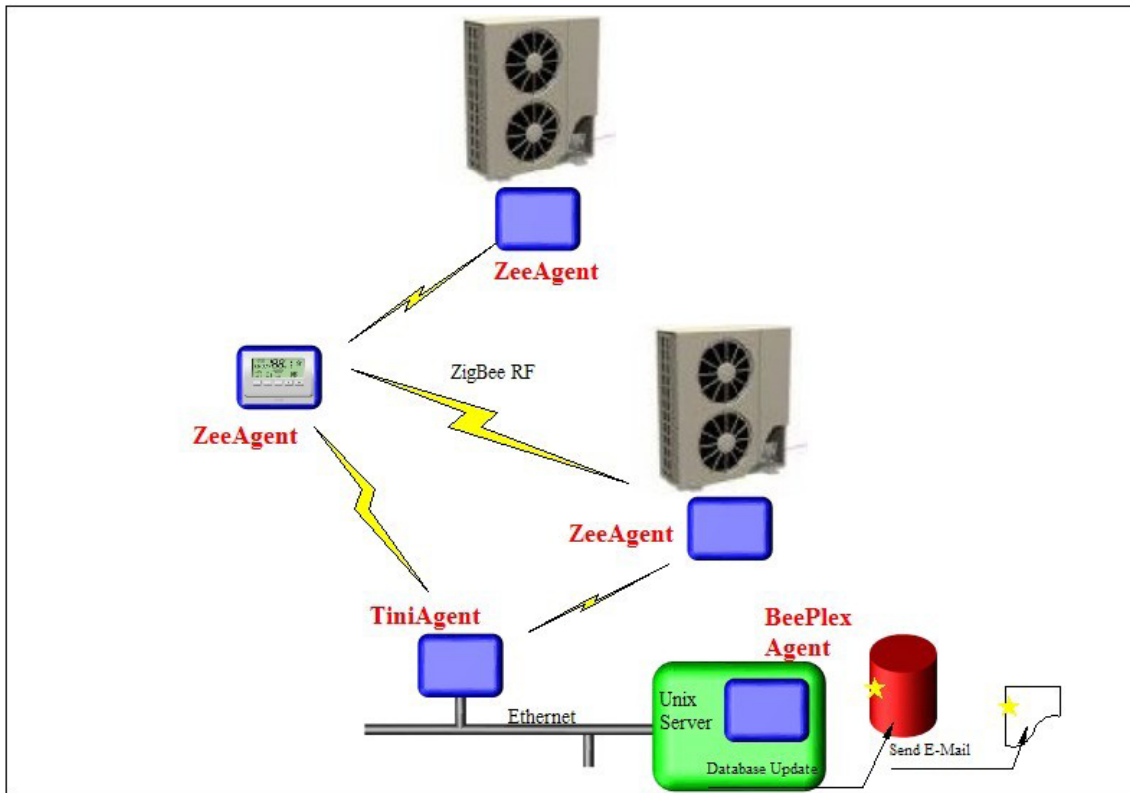
Sensing applications - Automated rule based actions that can be user modifiable using virtual services. Hosted by the sensor node. This is part of the application layer in the OSI model.

Tini Agent - Small, low-power devices that host software to provide a bridge to enterprise information systems.

Agents – EAI based services utilized ZeeAgents to enable communication with enterprise information systems. Agents are hosted by information systems that contain integration software which enables them to

perform their necessary functions. Agents enable automated notification services for enterprise email, automated voice notification and messaging.

Virtual Services – Console based task management interface, network monitoring and configuration.



### **3 *BeePlex Technology Overview***

The Enterprise Application Integration (EAI) technology that BeePlex is based on is in a very mature state almost to the commodity level. EAI has been folded into technology suites known as Web Services or Service Oriented Architecture (SOA). Virtually all large scale multi purpose software and hardware companies offer EAI technology within their product suites.

Initially EAI services were offered in stand alone products prior to consolidation into product suites.

BeePlex has taken a similar path, instead of folding EAI technology into a web based technology, EAI is integrated with Zigbee and IEEE 802.15.4. Although this is a unique application of the technology, the basic EAI techniques such as publish subscribe and persistent queue based messaging services have been in existence for 15 years or more.

In the late 1980's it was common to find IT systems with a disparate number of mainframes, midrange computing systems and personal computers with no easy way to integrate different applications on multiple platforms. Resources were needed to construct software programs and systems to translate data from the various source applications into a format the target system could use and understand. These programs were called adapters or interfaces. Cycle times and processing speed constrained each system. The mainframe batch paradigm was not acceptable to PC users. Mainframe users did not want to be constrained by the lack of output services, network bandwidth and legacy applications that did not exist on the personal computer platforms. In many cases the aging midrange system played the same role as the mainframe; both often ran proprietary networks that were difficult to connect PCs to.

The software technology BeePlex is based upon, Enterprise Application Integration or EAI solves the problem of management of the middleware point to point software connections by eliminating them altogether using a publish subscribe model rather than an adapter based architecture. Each sender publishes a "Topic" and each receiver subscribes to it. The topic is a similar to the subject line of an email, not the actual data itself. Software resources are not spent developing adapters for each connection. The sending node does not need to know the characteristics or state of the receiving node although full notification services are available where confirmation is necessary. It is essentially a fire and forget activity.

As message formats began to converge and mature the clear winner by the late 1990s became XML or Extensible Markup Language. BeePlex is ground breaking in the application of XML in the wireless sensor space. XML is a free and open standard. XML is used by virtually all large scale web content providers. It provides data exchange that is not tied to one specific technology or vendor. XML was designed to provide the quickest possible application processing for a data interchange format. By leveraging the advances in data interchange standards such as XML processing and mature EAI technology, BeePlex enables the development of large scale, enterprise wide sensor systems that can span large geographical areas and integrate with pre-existing municipal, state and federal systems.

BeePlex spans many levels of computing platforms in addition to ZigBee sensor environments. BeePlex has provided a messaging, publish subscribe backplane for distributed environments that include large capacity servers and mainframes. BeePlex has a software architecture designed from the ground up with scalability in mind. The application software readily scales down to the level of an embedded system due to the lack of rigid coupling and a component level architecture.

BeePlex is a modular component based Enterprise Service Bus framework that allows data synchronization at the enterprise level or to exchange data with business partners. Data integration is achieved by directly accessing the source and target application data stores, files, data bases or through the applications supported interfaces, be they CORBA, JMS, J2EE or whatever else is specified.

BeePlex provides faster completion times for e-Business integration projects including enterprise integration and B2B integration. BeePlex comes as close to a turn-key solution. The inherently modular architecture provides the advantage of a phased roll-out.

The publish subscribe model of business event handling means that the integration solution stays organized unlike point to point or stove pipe integrations that over time lend themselves to a spaghetti like deployment.

Applications programmers do not need to learn new programming languages and interfaces, scripting or other languages. There is no time consuming auditing, error handling, reporting or error correction programming tasks associated with BeePlex. Programmers who are integrating BeePlex with their current platform can concentrate on business logic alone.

BeePlex promotes the highest data velocity possible with the minimum of CPU, memory, storage and network overhead. The system as a whole tends towards zero latency.

BeePlex comes with a rich set of customizable application adapters for interfacing with common ERP, MRP, CRM and DBMS systems such as Siebel, Portal, DB2, Oracle and Remedy. Many integration tool vendors only offer adapters as separately chargeable items, often in excess of 100K per adapter. BeePlex offers its adapters as a part of the base product.

BeePlex adapters shield application developers from the complexities of the underlying messaging systems. They are free to develop programs using business logic and rules rather than spend time and resources on the intricacies of messaging [middleware](#) systems.

Each adapter is remotely configurable from one or more centralized management consoles. Users can configure as many adapters as needed in a flexible system of distributed agents or virtual machines. The agents can be started and stopped remotely or set to start up and shut down automatically with the host system. More than one agent can reside on a particular host and agent configurations can be readily migrated to other platforms or servers to effectively manage development in test and production IT environments.

BeePlex has replaceable, discrete components that plug-in, interact and intercommunicate while preserving their encapsulation and autonomy. BeePlex does not depend on the immediate availability of distributed components, brokers, remote servers, or other service providers in order to function. This is achieved with the use of a robust messaging system that queues messages using two phase commit and store and forward messaging procedures.

BeePlex is compatible with all major Relational Data Base Management Systems ([RDBMS](#)) and a number of legacy hierarchical data bases. With extensible adapters that are expressed as services it is possible to readily interface with most computing systems. This is especially important in working with public sector computing environments which often have proprietary applications for communications and operations. BeePlex is unique in the sensor software space in that it provides scalability up to the mainframe or large scale public sector computing environment while having adapter interfaces that can readily be adapted to legacy systems as well as predefined services for state of the art computing environments.

BeePlex has a mature monitoring and alerting system. The console has “traffic lights and auto refresh capabilities that enable the easy monitoring of agent status. Monitoring can take place at a single point of

control or can be distributed. BeePlex can be configured to send emails, pager messages, text messages to alert first responders and their support organization according to easily configured rules based notification.

The graphical user interface allows agents in the distributed network to be monitored and configured from one or more centralized location. All of this can be accomplished without additional programming tasks. The user interface provides a powerful set of auditing, statistical reporting, logging and remote system command execution facilities. Error events can be corrected in place via an easy to use editor and re-driven and re-routed on demand. Agents can be defined, configured and remotely started and stopped from the console. The GUI also provides an extensive set of context sensitive help and online documentation.

BeePlex uses TCP/IP Secure Sockets (SSL) for agent and console communications to enable the network traffic to be securely encrypted and authenticated. Object level authorizations restrict users according to security groups. Inter-agent communications are secured via the messaging provider vendor or standard.

BeePlex is quickly and easily installed via [InstallShield](#) multi-platform installation software, easing the installation tasks, which is of particular concern where there are large numbers of distributed hosts.

BeePlex has a flexible and rich set of customization features. Create re-usable filters, event processors, post processors or system commands. Trigger an application when a particular type of event occurs according to a rich set of user definable business rules. A scheme of substitution variables allows for great creativity in specifying BeePlex objects to make the integration system polymorphic according to the necessary context.

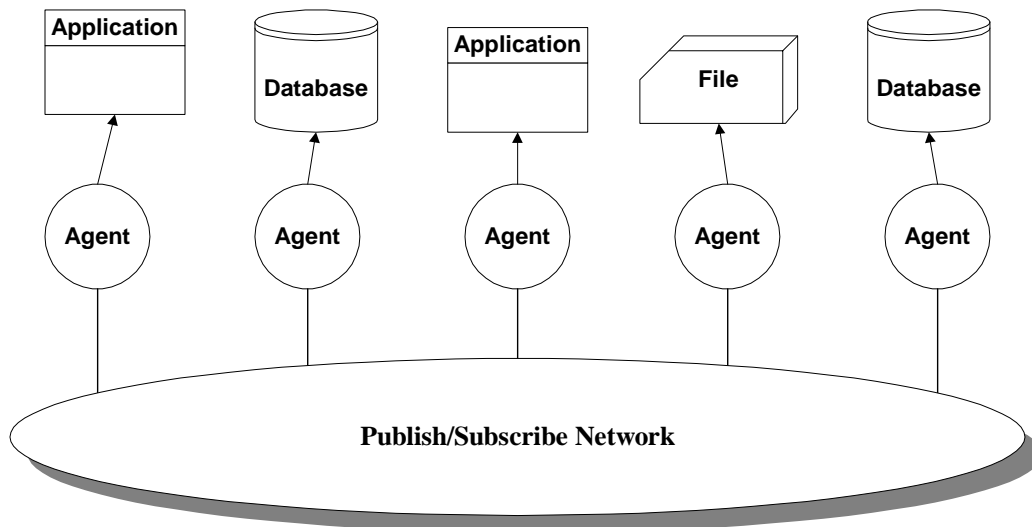
BeePlex integration architecture is grounded in the J2EE standard messaging API, the Java Messaging Service (JMS) see <http://java.sun.com/products/jms/>) This provides BeePlex with an asynchronous messaging middleware interface with complete vendor independence.

The [JMS](#) backbone of the BeePlex enables all event messages exchanged on the message bus are text messages in an XML format. This provides ease of mapping and platform independence and support for virtually all web browsers.

The publish subscribe model is used along with documented enterprise topic streams that reflect business processes in an abstracted form.

A leading customer reported that the underlying technology BeePlex is based upon enabled a estimated five year integration project being completed in only seven months.

### **3.1 BeePlex Architecture**



**Figure 1. Modular Integration Architecture**

BeePlex is process-driven, rather than data driven. BeePlex operates using a network of distributed agents that communicate using the Java Messaging Specification (JMS). It eliminates the complexity of point-to-point integration and allows for rapid deployment with a moderate learning curve.

### **3.2 The primary features of BeePlex**

- ◆ **Flexibility** – the solution is cheaply and easily customized or re-organized
- ◆ **Modularity** – a successful solution cannot be monolithic, therefore BeePlex has replaceable, discrete components that plug-in, interact and intercommunicate
- ◆ **Asynchronosity** – your integration solution must not depend on the immediate availability of distributed components (brokers, remote servers etc.) in order to function. This is achieved with the use of a robust messaging system that has two-phase commit and store-and-forward messaging capabilities
- ◆ **Extensibility** – BeePlex has a natural, well-defined growth path. As the use of any system grows, and the number of tasks it performs multiplies, the internal complexity of the system increases. BeePlex is able to absorb new requirements with the minimum of impact on the existing infrastructure and to accommodate more tasks with relative ease
- ◆ **Scalability** – as the solution is extended (or the volume of usage increases) the impact to the BeePlex system is linear in fashion, without bottlenecks or exponential use of resources
- ◆ **Reliability** – BeePlex is robust. It has no single points-of-failure and functional fail-over can be easily implemented
- ◆ **Performance** – BeePlex promotes the highest data velocity possible, with the minimum of CPU, memory, storage, and network overhead. The system as a whole tends towards zero-latency.
- ◆ **Simplicity** – the BeePlex solution is uncomplicated and elegant

## 4 Background and Related Research

According to [Moore's Law](#), an empirical observation made in 1965, the number of transistors on an integrated circuit for a minimum component cost doubles every two years. This is attributed to Gordon Moore, the founder of Intel corp. Recent computer technology roadmaps do not dispute this metric at least for the next three generations. It is widely accepted as an approximation or benchmark to understand that integrated circuits such as ZigBee SOC technology capability will increase at orders of magnitude within the normal product cycles of hardware design and application software development.

[Metcalfe's Law](#) concerns the value of a communications network, like a ZigBee mesh network, is proportional to the square of the number of users, or in this case ZigBee nodes on a system. This was developed by Robert Metcalfe, a pioneer in Internet and ZigBee technology. Metcalfe's law originally applied to the Internet and World Wide Web. An example of how the value proposition works is that a single fax machine is of no utility. Fax machines have value because of the totality of people that can potentially send faxes to one another. If a small exclusive group held fax machines they would not have much in the way of value. There has been debate about the degree of value held by Metcalfe's Law but no one questions that there is an intrinsic increase in value of a network as it increases in nodes.

In the case of a municipal ZigBee network spanning a significant area the value proposition grows remarkably. Without publish subscribe technology the ability of such a network to provide value and function is constrained by the bandwidth and ability to process the massive amount of data such a network would generate in any but the smallest disaster scenario.

The research developed this proposal is at the intersection in both Moore's law and Metcalfe's Law. As the capabilities of ZigBee SOCs increase due to Moore's law the size and value of networks increase according to Metcalfe's Law as it becomes more feasible in terms of economics, size and power to have larger ZigBee networks. We are now proposing ZigBee networks to span geographical areas this size of major metropolitan areas. The value of a network such as this cannot be overestimated as long as the data payload can be delivered and processed reliably.

The ZigBee SOC technology itself has matured enough to make the cost of a large ZigBee network feasible (Moore's law). BeePlex technology has been developed and will continue to mature as the value of the network increases by large scale enterprise level deployments (Metcalfe's Law).

In conclusion, BeePlex is commercially available on ZigBee compliant platforms to provide a way of reliably delivering sensor payloads especially in scenarios where a mesh network can be disrupted due to damage or disaster recovery.

## 5 About SolutionsWorks

SolutionsWorks is a privately funded Oregonian company. It was founded in May 2005 with the vision of creating reliable, robust, and - most importantly - usable technology. We like to say that, in a word, we strive to create *elegant* solutions.

We aim to apply technology to niche areas that have not traditionally been associated with high-tech in the past. SolutionsWorks is uniquely positioned to assimilate the full range of new low power and inexpensive wireless ZigBee networks into the enterprise computing environment.

For more information please email [info@solutionsworks.com](mailto:info@solutionsworks.com)

Copyright © 2007 SolutionsWorks Inc.

Corporate Headquarters:  
SolutionsWorks Inc.  
2828 SW Corbett Ave. #117  
Portland, OR 97201  
503-343-9984  
Fax 415-358-5889

## **5.1 Links**

[Publish Subscribe](#)

[MQSeries Publish/Subscribe Applications](#)

[A Java-based Publish/Subscribe Middleware](#)

[Publish-subscribe model connects Tokyo highways](#)

[OMG Data Distribution Service: Real-Time Publish/Subscribe Becomes a Standard](#)

[A Java-based Publish/Subscribe Middleware](#)

[An Approach to Model and Validate Publish/Subscribe Architectures](#)

[Striving for Versatility in Publish/Subscribe Infrastructures](#)

[Adaptive real-time publish-subscribe service model for mobile communication environments](#)

[OMG Data Distribution Service: Real-Time Publish/Subscribe Becomes a Standard](#)

[Event-Driven Architecture vs. Publish-Subscribe Systems](#)

[Developing Applications on the Publish-Subscribe Model](#)