

When Should I Customize and How?

Christopher Johnson

2010

A large, 3D green arrow pointing upwards and to the right, symbolizing growth or progress. The arrow is positioned diagonally across the frame. At the base of the arrow, the year '2010' is written in large, 3D green numbers. The background is a solid blue color.

Needs



- Format/Validation changes to existing screens/fields
- Additional information storage
- Minor functionality change to existing process
- Major functionality/process change
- New processes

Analysis



- Research/ask about existing software capabilities
 - Make sure functionality does not already exist
 - Is it configurable through the user interface? (MS CRM – New entities for example)
 - Does a newer version of the software include the desired functionality?
 - This is important for future upgrades as well as current needs

Analysis



- Cost/benefit analysis
 - How much is it worth?
 - How much time does it save vs. the time to code?
 - Is an extra mouse click worth the price?
 - Benefit beyond time savings?
 - Protection of data
 - Security
 - Storage /Manipulation of additional data

Analysis



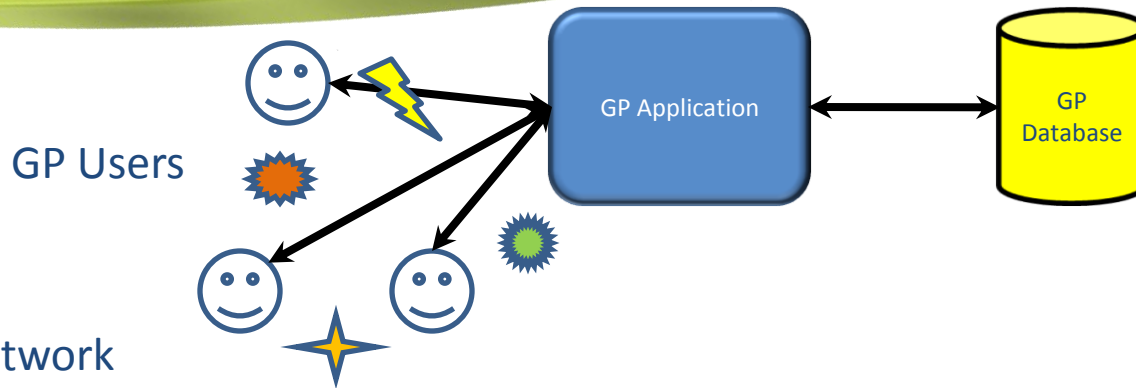
- Process Review
 - Can existing processes be modified to minimize programming efforts and development time?
- System Review
 - Analyze flow of data
 - Separate (mentally) application functionality and data
 - Looking at the “lifecycle” of specific data elements can help to get a better understanding of the next best steps
 - Analyze application functionality
 - Determine what information is being stored (above) vs. what is being done to it by the users through the application

Different Levels of Customization

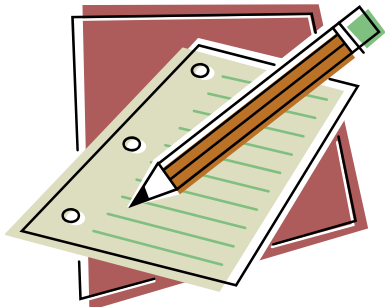


- “Native” Changes
 - Small screen/field format, validation, accessibility
 - Modification to existing processes/functions
 - New screens integral to existing process
- Integration with other applications
 - ISVs or “off-the-shelf” solutions
- Custom built application to “Fill the Gaps”
 - Interactive through APIs (preferred)
 - Data integration with back end procedures (i.e., e-connect)
 - Data integration through direct database calls (boo!)

Custom Built Application Case Study



Outside Network



Customization Options



- Customize “Native” GP?
 - Requires Citrix/Terminal Server for access
 - Requires GP Licenses for each user
 - Dexterity or VBA client side coding
 - Process is outside GP standard functionality
 - Requires additional consideration during upgrades
- Submit files/spreadsheets?
 - Difficult to control
 - Multiple versions
 - Not automated
 - Asynchronous (batch process)
 - Not cool

Customization Options



- Custom Web Application
 - Must get information into/out of GP
 - Real time
 - Integration options
 - User maintenance/authentication
 - Control/validation

Integration Options



- One data source with a real-time interface
 - Best option
- Two data sources with “one way” integration
 - Second best option
 - One is “read only” to users
 - Updates only come from other data source

Integration Options



- Two data sources with “two way” data integration
 - Viable option but costly
 - Can be complex
 - Can require extensive programming and maintenance
 - Synchronization issues can be difficult to track down and irrevocable
- Two data sources, no integration
 - Worst Option
 - Double data entry
 - Data out of synch
 - Madness ensues

The Hard Way



- Write application or database programs to directly insert/update/delete information to/from GP database
 - Figure out tables/relationships in database
 - Write insert/update/delete procedures with error handling
 - Security
 - Time consuming
 - Ugly
 - Error prone
 - Not supported by MS
 - But sometimes you gotta do what you gotta do!

What are GP Web Services and Why Do I Care?



- NOT a Web Page!
- NOT a Web Page!
- NOT a Web Page!

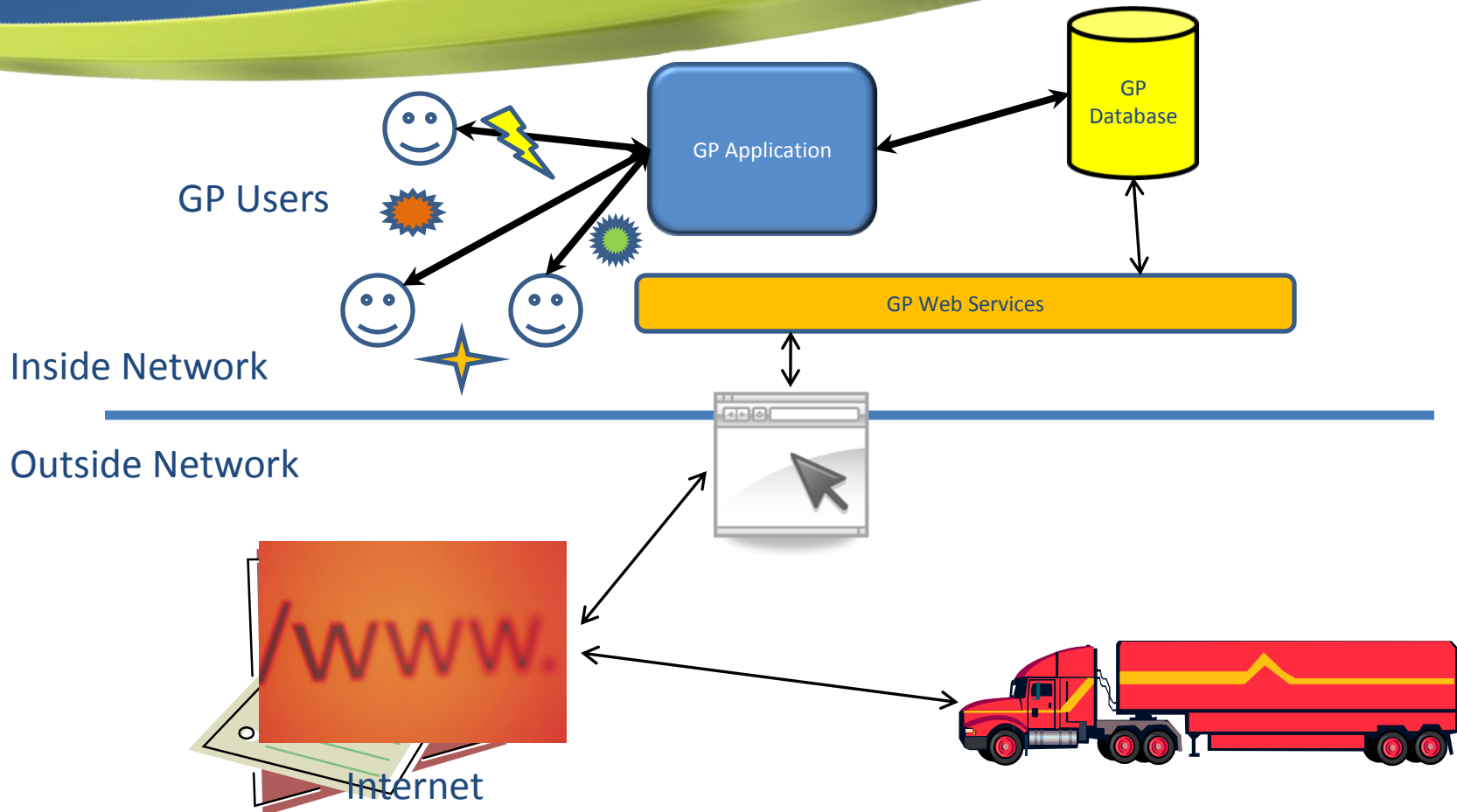
What are GP Web Services and Why Do I Care?

- GP API (Application Programming Interface)
- A set of programs which allows a programmer to interact with an application using some standard agreed upon method accessed via HTTP (web)
- Service Oriented Architecture
 - Trust me, don't worry about the details, order from the menu, and eat what you are served.
 - Stay out of my kitchen!
- Pre-built, pre-defined processes that handle the back-end data, processes, and exceptions
- Direct hooks into GP from .NET
 - Actually it uses E-Connect which is a set of stored procedures that talks to GP
- Uses XML
- WSDL?

WSDL

- Web Services Description Language
 - XML-based language for describing web services and how to access them
 - Provides a standard, consistent menu/description of services
 - Gets
 - Sets
 - Deletes

GP Web Services API Solution



Questions?



Christopher Johnson

cjohnson@socius1.com

(440) 717-3230