

# Building Business Systems without Programmers?

---

A white paper describing an automated system development tool which will change the way we build custom business systems

Author: John Bryant

March 2008

**Extract:** *Programming has been an issue for years. It has always been a major cost in any system development project. In an attempt to reduce the cost, the trend in recent years has been to outsource, or to set up an off-shore subsidiary where wage rates are low. This can reduce costs but is not without its difficulties....  
With Pyinna you reduce the need for programmers by around 80%.*

---

## Building Business Systems without Programmers?

---

**A tool that eliminates 100% of programming from systems development is unlikely to be available until we reach ‘The Singularity’, predicted to be around 2030<sup>1</sup>. Until then a human programmer will be needed to deal with the custom logic required in most business systems.**

**However, if you were to imagine a tool that gets close to removing the need for programming from business systems development, the tool described in this White Paper would be it.**

**We describe a software tool that automates business systems development, removing approximately 80% of the conventional programming – and it is available today! It will change the way we build & implement business systems for a long time to come...**

---

Picture this – you are a business analyst or manager of a department of analysts. You are responsible for delivering IT business systems for your company – solid, reliable, really cost effective solutions, ones that users feel are a benefit to their jobs. You have a stack of requirements on your desk, some are pressing...

There is that new project which needs a system to coordinate the team, track tasks and milestones, report progress, keep track of budgets...

There is the problem of replacing a plethora of Excel files around the business with a database system which enable users to share data – this is a knotty one, you know users won’t give up their pet spreadsheets easily...

There are a couple of key systems, core to the business that are creaking with age. You have patched them many times over the past year or two but they are now really struggling and must be replaced soon – how do we do this with minimum disruption to the business – this is high profile...

And that’s just the start!

---

<sup>1</sup> Within a few decades, machine intelligence will surpass human intelligence, leading to **The Singularity** -- technological change so rapid and profound it represents a rupture in the fabric of human history. The implications include the merger of biological and non-biological intelligence, immortal software-based humans, and ultra-high levels of intelligence that expand outward in the universe at the speed of light. The Law of Accelerating Returns by Ray Kurzweil

What are your options for these 3 projects?

For the project tracking system you could buy a standard package off the shelf, set it up and run training courses for users – but... there are some quite special requirements – rather more than a standard package could handle. Maybe you could create something specific in Excel – everyone knows how to use Excel. Wait a minute though, won't that just exacerbate the 'plethora of spreadsheets' issue... Maybe we'll have to build something custom.

For the spreadsheet replacement project – that's big – there are a lot of groups of users, a big data import issue, the biggest issue is lead-time. You are probably going to have to outsource this – you can handle the requirements-definition in-house and the project management but the programming you'll have to put outside. You will have to select a partner carefully - Users won't want to spend time defining requirements, and then wait months for a system which typically will need many iterative revisions before they can see their data and begin to use the system – there is risk here...

For the core systems replacement, you will probably need to find a specialist software development partner – this will not be cheap, there will be a lot of programming involved. The work to build the user-requirement specification, vendor selection, partner management, project management, data migration, rollout, change management – not to mention the budget and the potential disruption to the business - this will require involvement and sanction by the Board. This will mean a big budget and timescales; you are talking a year plus, per system maybe a lot more.

The biggest issue is the programming. Defining the requirements so tightly that the techies doing the coding understand precisely what you want – but it's never like this – there are always communication issues: users to analysts; analysts to programmers; programmers back to analysts and users. This causes time delays, cost overruns, user frustration and potentially user acceptance issues even project failure.

Wouldn't it be great to short-circuit this... to maybe eliminate most of the programming ... this would mean delivering the system in much less time and at a fraction of the cost. Could this really be possible, what sort of a development environment would be needed to do this? To be able to implement systems with significantly less programming - where could you find a tool to deliver this dream...? Actually one does exist.

---

**Pyinna is a development tool that automates business systems development, reducing programming by around 80%**

**Pyinna is a development tool for business analysts, not programmers.** The business analyst can control the majority of the process from user requirement through to system rollout – and in a fraction of the time typically seen using conventional development cycles. We are talking here about delivering the initial

core system back to users in days or in some cases hours after agreeing the requirement with users. This is amazing. The users will see their data, their screens, and start using the system immediately. Removing the usual frustrations of non-involvement, long time delays between promising a shining new system and its eventual delivery.

**Pyinna is a development tool that automatically builds big as well as small systems.** In deed, you can build very large systems with Pyinna: vast SQL databases serving thousands of users, across regional or global boundaries – there is no practical limit. This is a development tool for seriously sophisticated business systems.

Major systems that would traditionally take years to implement can be built and deployed to users in months or even weeks and at a fraction of the cost. What's more, these systems (because they are built using Pyinna) will be solid and stable from day one. And, the cost of annual licences and support & maintenance contracts is substantially reduced because most of the system, if not all, is supported in-house.

**With Pyinna the advantages simply pile up.**

Developing your system using Pyinna, the need for programmers is substantially reduced. Many system expansion requirements can be implemented by the analyst with minimal involvement of programmers.

Programming has been an issue for years. It has always been a major cost in any system development project. In an attempt to reduce the cost, the trend in recent years has been to outsource, or to set up an off-shore subsidiary where wage rates are low. This can reduce costs but is not without its difficulties – communication can be difficult; project managing at a distance is hard; there is far less control over quality, timescales and budgets – cultural friction often becomes a management problem. With Pyinna you reduce the need for programmers by around 80%. So now you can easily afford to employ local people.

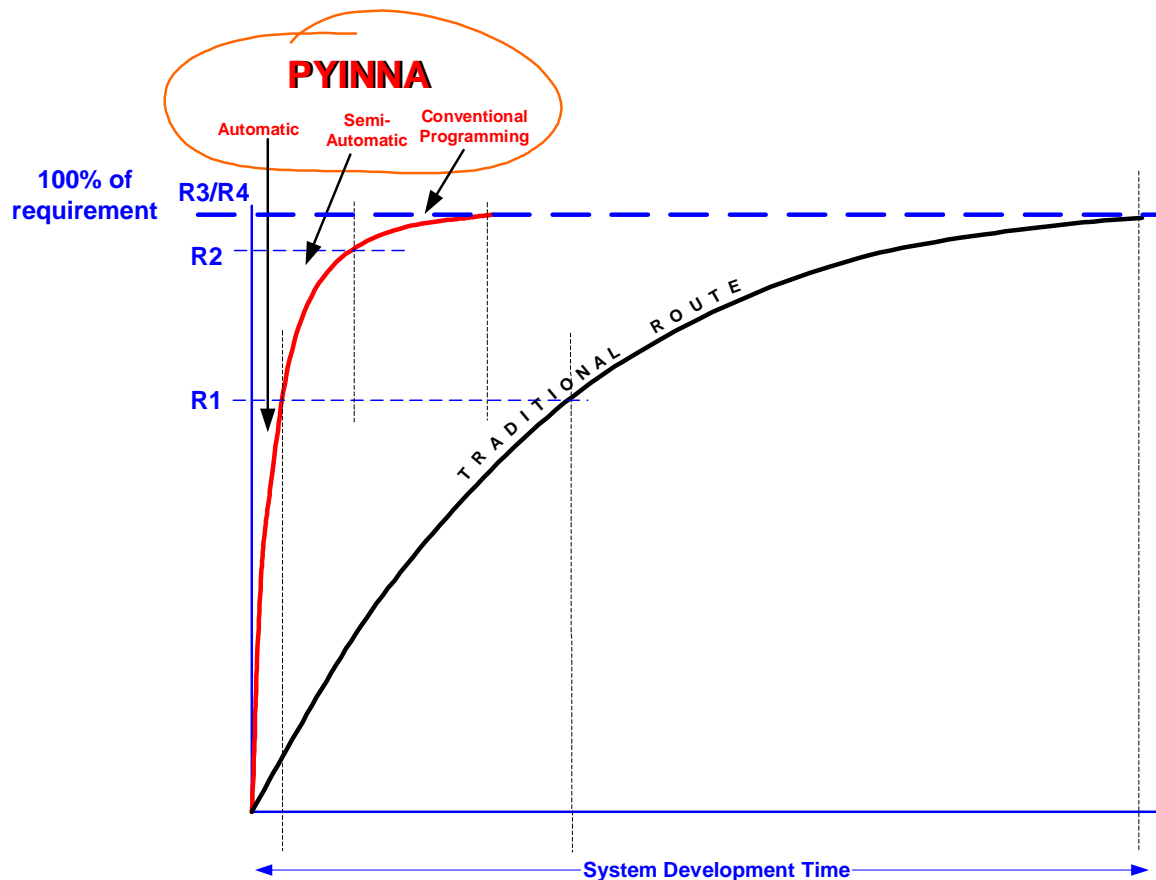
**What knowledge do you, the business analyst, require to develop his system using Pyinna?**

1. Your existing core skills of defining business processes including data and process modelling
2. A working knowledge of Microsoft Excel
3. A working knowledge of the Pyinna Macro Engine – acquired during initial on-the-job training where Pyinna support team configures the system with the analyst/s to get them started.

That's it!

**PYINNA delivers Huge Productivity Advantages over the traditional system-development route**

Consider the following diagram – the BLACK curve depicts the traditional programming route using tools such as C#, .net, SQL Presentation Suite; VB, Visual Studio etc; compare that with the RED curve which depicts the PYINNA route to achieve the same user requirement...



**R1** - Represents the proportion of the system requirement (the core system; database; user-screens; search & listing facilities) automatically built by PYINNA. *Notice that the traditional (programming) route takes about 10 times longer!*

**R2** - Represents the proportion of the system requirement (the core system plus the user processes and navigation created by business analyst using PYINNA semi-automatic tools

**R3** - Represents 100% of the requirement using PYINNA plus some conventional programming for specialised elements.

**R4** - Represents 100% of the requirement using only the traditional route – without the benefits of PYINNA

**So how does Pyinna do it?**

At the heart of Pyinna is the Process Engine. The Process Engine is deployed in two key ways: -

- (a) For automatic building of the core system (database, screens, fields, buttons, tabs, menus, Query-By-Example searching, and result-set lists presentation)
- (b) As a semi-automatic customizing tool for building the business processes and navigation as defined by the business analyst

**The step-by-step process of automatic building of the core system: -****Step 1**

Develop the user / business requirements specification

**Step 2**

From the user / business requirements specification - define the data model and field formats using MS-Excel templates provided by Pyinna.

**Step 3**

Import the completed Excel spreadsheets to Pyinna – and initiate the auto-build process.

**Step 4**

The business analyst can now log in to his newly built system which will reflect exactly what he has defined. If he needs to make changes these are made firstly in the Excel file – this is then imported again and Pyinna rebuilds the system to include the updates.

**Step 5**

The analyst can now load user / business data to the system using Excel user-data import templates provided by Pyinna

**Step 6**

Once the business analyst is comfortable with his system, this can immediately be released to users who can log-in via browser over Intranet or Internet as required. They will be amazed to see the screens they have asked for along with their data and ability to search for records, add new record, edit records, list result sets from complex queries – in fact start using their new system. Now that's impressive!

**Step 7**

Build advanced user / business processes

**Step 8**

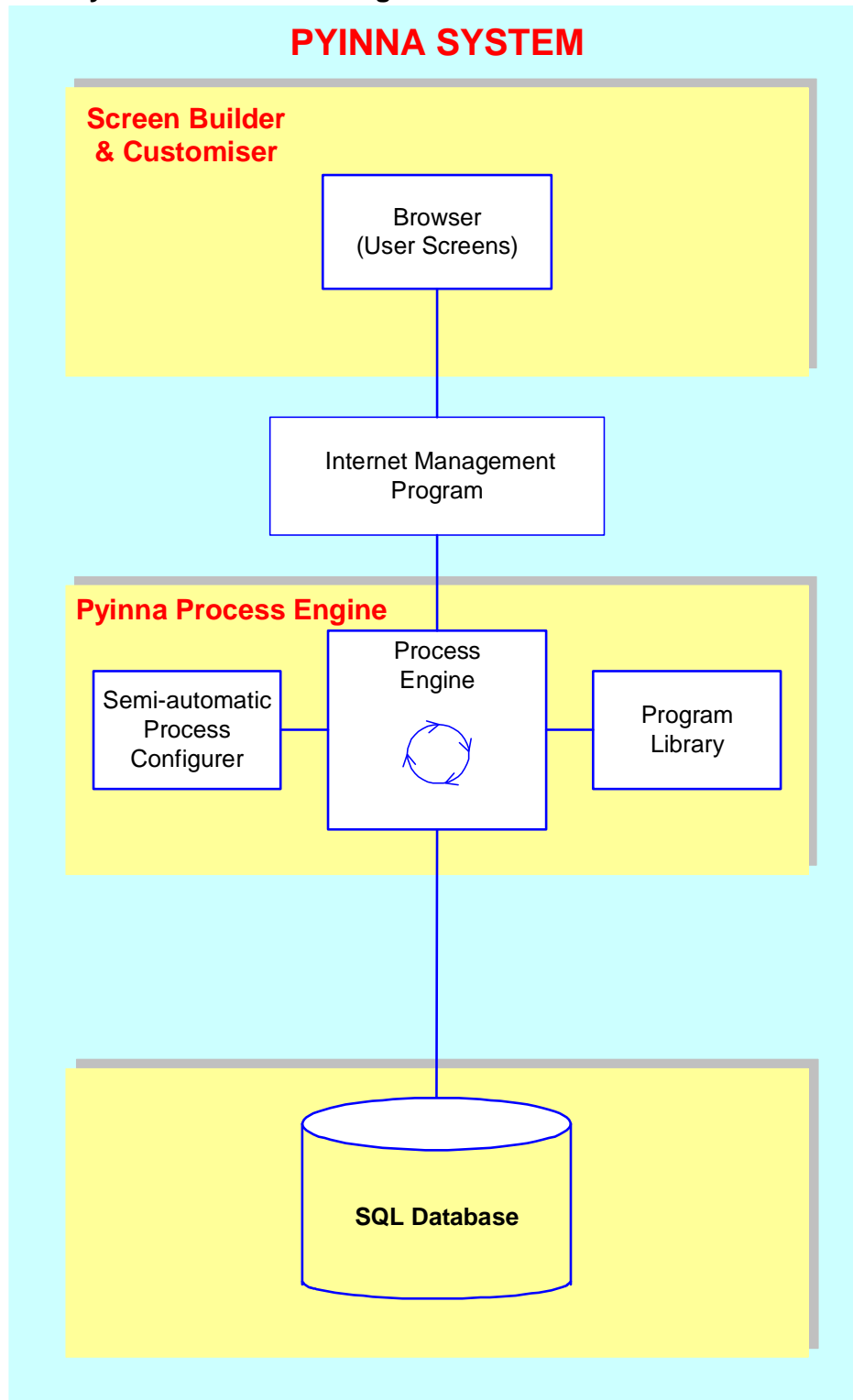
Set up rights for user-groups and individual users

**Step 9**

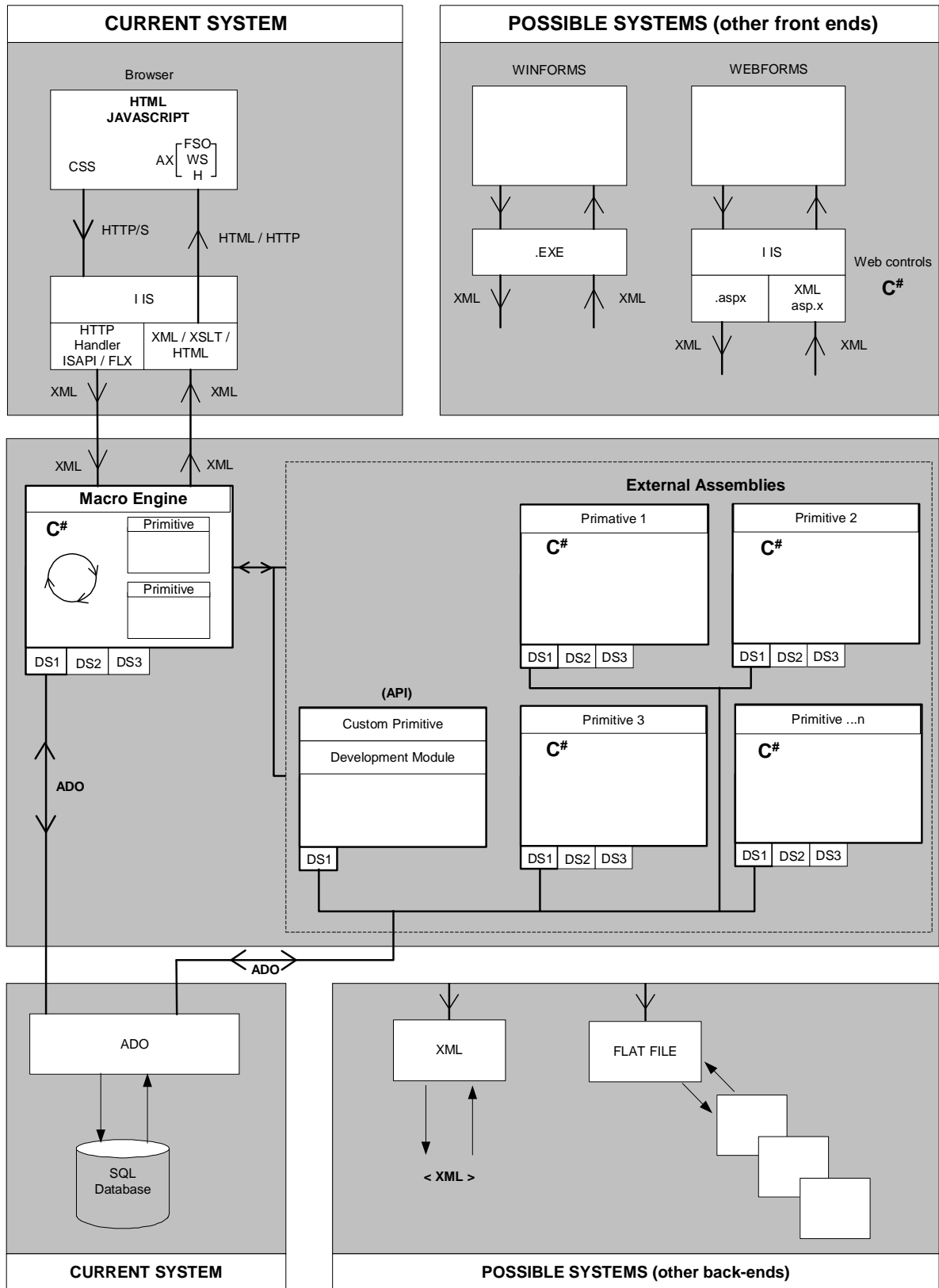
Rollout to entire user community

Let's now look at the structure of the Pyinna system...

Pyinna architecture diagram-1 for non-technical readers



**Pyinna architecture diagram-2 for technical readers**



**PYINNA core capabilities: -**

- **Automatic database creation**  
Pyinna will automatically generate the entire relational database schema: tables; one-to-many & many-to-many links; pick-lists; fields (types, formats, validation) – by inputting from spreadsheet you are documenting the system as you go. Pyinna will automatically create tabbed list areas for showing dependant data (for example: if you build a quote table and a quote-line table, a quote lines tabbed list will automatically appear on the quote screen)
- **Automatic user screens creation**  
Pyinna will generate users-screens in Web-browser environment automatically, fully reflecting the generated database schema
- **Automatic query-by-example search facilities**  
Pyinna will generate QBE (query-by-example) searching facilities automatically to enable users to search on field combinations and list result-sets
- **Tabs & Menus**  
Pyinna automatically generates tab-screens and standard user-menus
- **Style Sheets**  
Pyinna provides a selection of different style sheets to give a different look & feel to the user screens – you can create your own system sheets to reflect in-house styles
- **Custom Screens**  
Pyinna includes a screen customizer which provides facilities to create your own user screens
- **Standard Browser deployment**  
As soon as Pyinna has created your system, it can be deployed to users straight away – users can access via the web or using the company intranet using standard browsers such as MS Internet Explorer
- **User navigation**  
Pyinna automatically sets up optimum navigation between screens it has automatically created
- **Business processes**  
The Pyinna process-engine and macro language provides the Business Analyst with very comprehensive facilities to build business processes (see the section below: 'Building Business Processes')
- **User rights management**  
Pyinna provides facilities for setting up user rights and authority at user-group or individual level

- **MS Office linkage**  
Pyinna provides transparent links to MS-Word (including output to merge templates) and Excel (for both imports and reporting)
- **Links to external data sources**  
Pyinna APIs provide facilities for linking to external SQL databases.
- **Harness modern technology – tried & tested platforms**  
Using Pyinna to develop your system ensures that you will be completely protected from software compatibility and similar technical issues
- **Easy data import**  
Pyinna's data import manager provides all the data mapping and importing facilities for migrating data from the variety of sources around your company

### **Building Business Processes**

All commercial business processes can involve any or all of the following five process-attributes which when combined reflect the business process: -

1. Sequential Flow
2. Conditional Flow
3. Approval loops
4. Nested data sorting and branching
5. Data manipulation

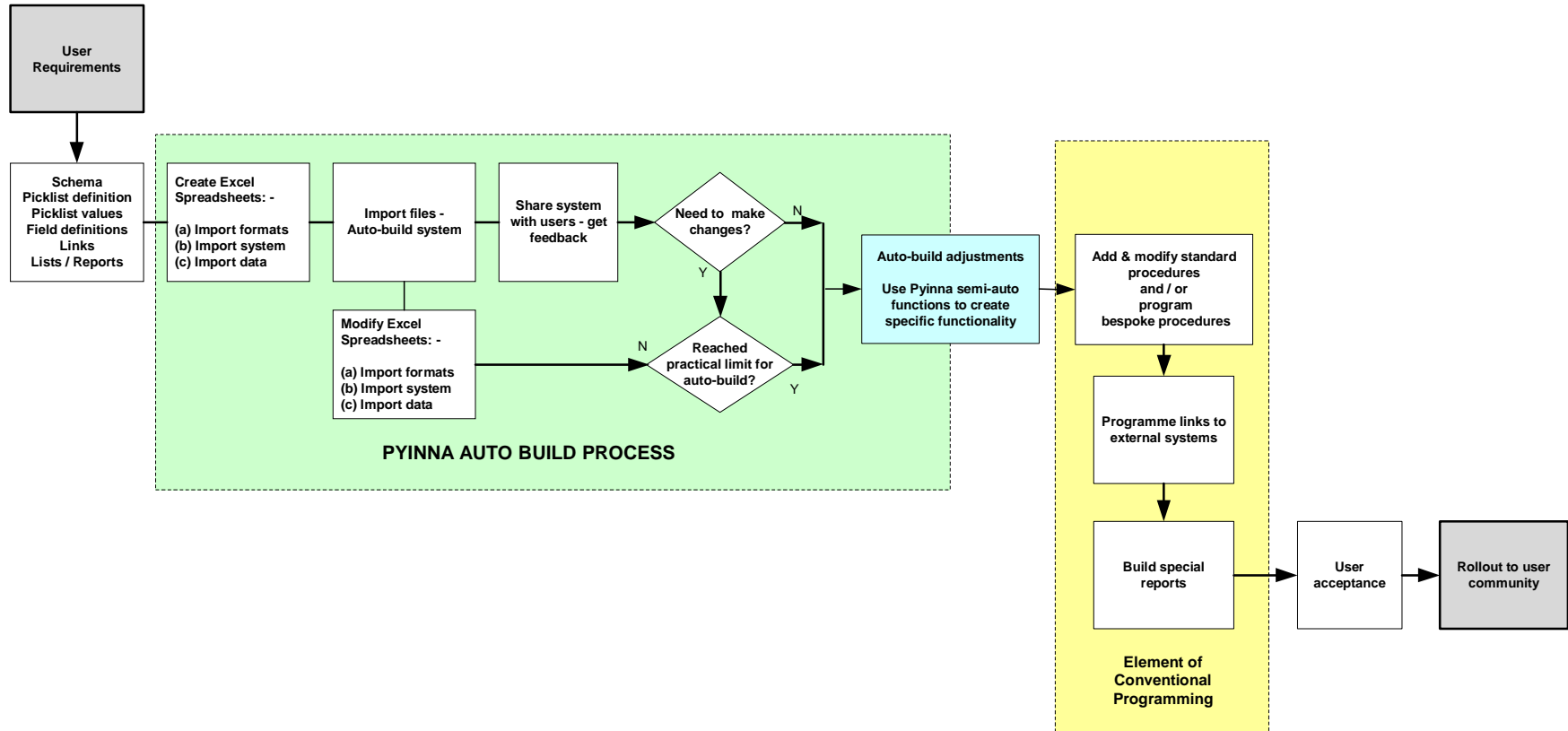
Pyinna manages these process attributes through its unique Process Engine, at the simplest level the Process Engine manages: -

- Sequence of steps
- Branches
- Loops

The process engine is at the heart of the Pyinna system as shown in the architecture diagrams (pages 7 & 8). The process engine, primitives-library (program library) and macro language together form a uniquely powerful system for generating both the core-system automatically and as a business process creation tool. The primitives-library contains a set of process templates (primitives) which are strung together in any combination to create the business process required.

The approach to building a business system is outlined in the following diagram:

## Building business systems using PYINNA



## Compared to conventional business systems development Pyinna delivers major productivity advantages

To illustrate the productivity gains Pyinna delivers we will take the example of adding a new requirement to an existing business system; a Time-Sheet system.

Your users want a new button added to the Time-Sheet system. The new button is to be added to the main time-recording screen to replicate a previous screen, so the users do not have to start from scratch each time, populating fields containing the same information.

Your normal action for dealing with such a requirement is to contact the support team (external vendor or internal programming team) to get them to quote for the work. The following programming quotation is submitted for your approval...

### Quotation to add the new button using conventional programming

Step	Programming task	Minutes
1	Preparation and Analysing Requirement	60
	Add Button to an Existing Page to link	60
	Create a New ASP Page and add to SourceSafe	60
	Receive POST Data from an Existing Screen	20
	Handle errors	20
2	Connections Handling: -	40
	• Get Connection String from config	0
	• Create Connection to Database	0
	• Test Connection is successful	0
	Handle errors	40
3	Create Command Objects	0
4	Create Result-set Objects	0
5	Create a Screen to select record	120
	Bind Screen to Database table	0
	Handle errors	40
	Display screen to user	0
	6	Receive POST : selected item from web
Data-grid to hold result-set and binding to DB		60
Identify fields to be copied, where is this coming from		60
Clone record-set		0
For each record Item:		
Generate Unique Ref for new item (this involves adding prefix to, etc)		60
Change the parent of the selected item		0
Reset the fields to be not copied		0
Update table with cloned record-set		0
Handle errors		40
7	Create screen to display message	60
	Display Confirmation screen	0
	Refresh the current window	60
8	Recompile project and fix problems	120
	Test the process	120
	Version control	0
	Deploy process for user to test	0

**1060 minutes**









This is equivalent to 2.5 normal programmer-days.

Compare this to the same task using Pyinna which takes just 10 minutes!

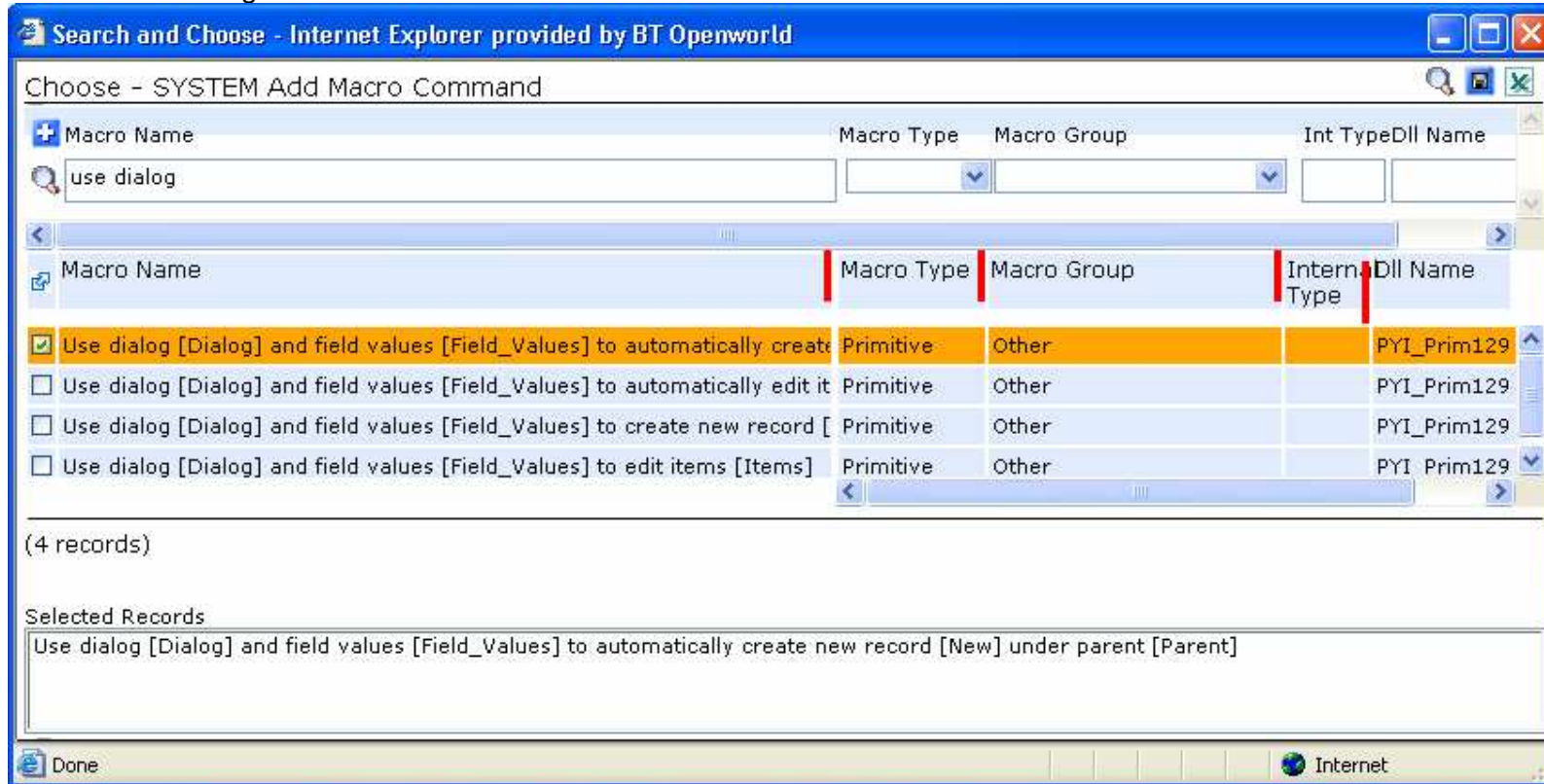
The next series of screen-shots are taken from Pyinna as *it* adds the button...

The simple procedure using PYINNA to add the new button to the Time-Sheet system...

STEP 1

Line	Process	 What you do in PYINNA
1	<u>Establish the record</u> that the user is focused on	Get context <u>Current TSheet</u> of <u>Window</u> of type entity ( <u>Tsheet Week</u> )
2	<u>Show a screen</u> for the user to choose a previous record	Choose multiple <u>Selections</u> using list <u>List (Tsheet Week For User [PUBLIC])</u>  filtered by <u>Filter</u> (attributes: MODE=single) <u>edit attributes</u>
3	<u>Get all of the lines</u> associated with the previous record the user has chosen	Get results <u>Previous Lines</u> using list <u>List (Tsheet Lines For Tsheet Week [PUBLIC])</u>  filtered by <u>Selections</u> <u>add attributes</u>
4	<u>Loop</u> through all the line records...  ...until we have done them all	  TSheet Line Copier Looper <u>Previous Lines</u> and <u>Current TSheet</u>  If <u>Previous Lines</u> empty then do command  Exit this loop <u>info</u>  Otherwise  Move single item from <u>Previous Lines</u> and add into <u>Previous Line</u> <u>info</u> <u>add attributes</u>
5	<u>Get only the fields</u> we wish to have copied to the new records (ignoring hourly info)	Get field values <u>Field Values</u> for record <u>Previous Line</u> using dialog <u>Dialog (Tsheet Line For Copying)</u> 
6	<u>Create</u> New lines connecting them to the record the user is focused on	Use dialog <u>Dialog</u> and field values <u>Field Values</u> to automatically create new record <u>New</u> under parent <u>Current TSheet</u>
7	<u>Clear</u> the reference to a previous line (so that it can be filled later with the next line to be copied)	Clear everything in <u>Previous Line</u>
8	(unseen) <u>Refresh</u> the browser (so that the lines are visible)	

STEP 2 – selecting the ‘create button’ Primitive



**STEP 3 – Adding the new button to the screen**

The screenshot shows the Flexi.NET Designer interface for editing a list. The main window is titled "Editing List: Designer - Internet Explorer provided by BT Openworld". The "Designer" area shows a list configuration for "Tsheets Week Details" with a width of 900 and height of 570. The list has tabs for "Tsheets Week", "Details", and "TSheets Line". The "Tab Name" is "Tsheets Week".

Below the tab settings, there are checkboxes for "Fixed Width?" (unchecked) and "Label on Left?" (checked). The "Add to Column Grid" is set to 1.

The main table lists the fields for the list:

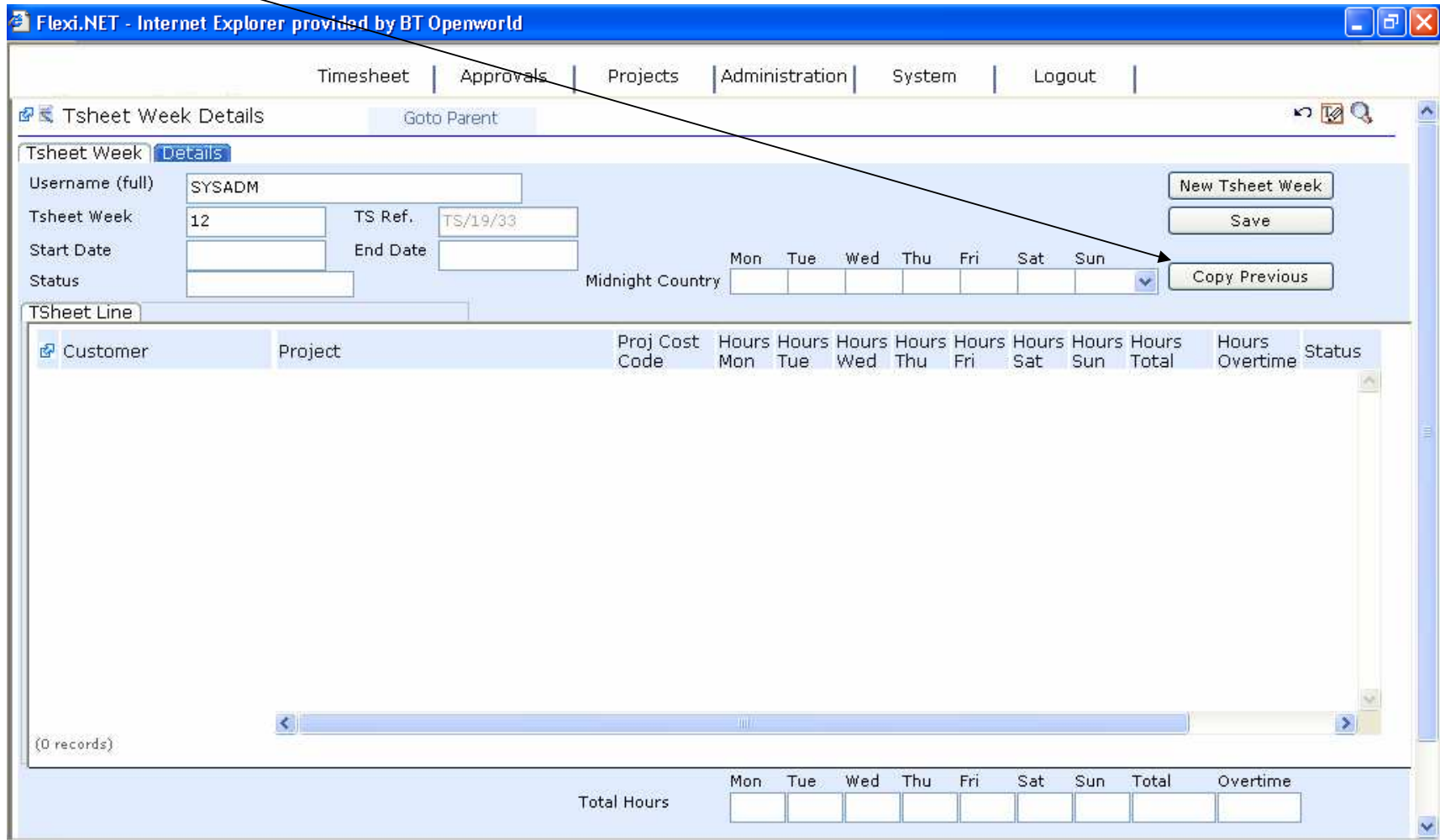
Field Name	Left	Top	Height	Width	Editable?	Mandatory	QBE field?	Transparent	Visible?	Add Flag
Tsheets Week	ta									
Copy Previous	bt	820	85	22	120					
Username (full)	lal	8	22	22	120					
Username (full)	at	120	22	22	240					
Tsheets Week	at	120	46	22	100					
Tsheets Week	lal	8	46	22	120					
TS Reference	sy	300	46	22	100					
TS Ref.	lal	240	46	22	120					
Start Date	dr	120	70	22	100					

A "System Button Command" dialog box is open, showing the following configuration:

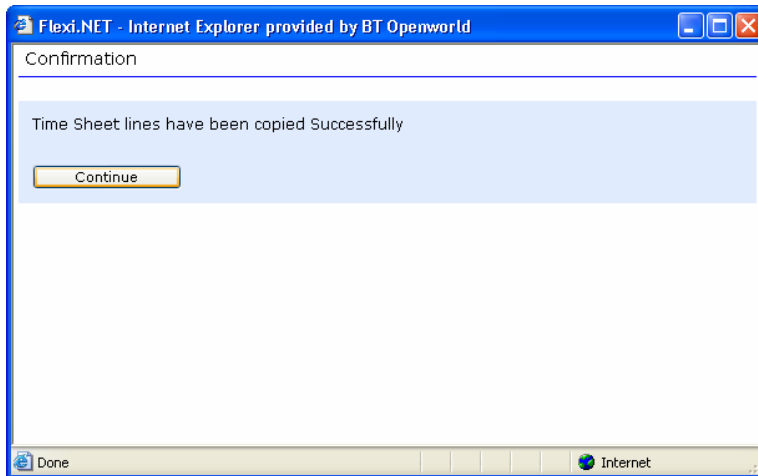
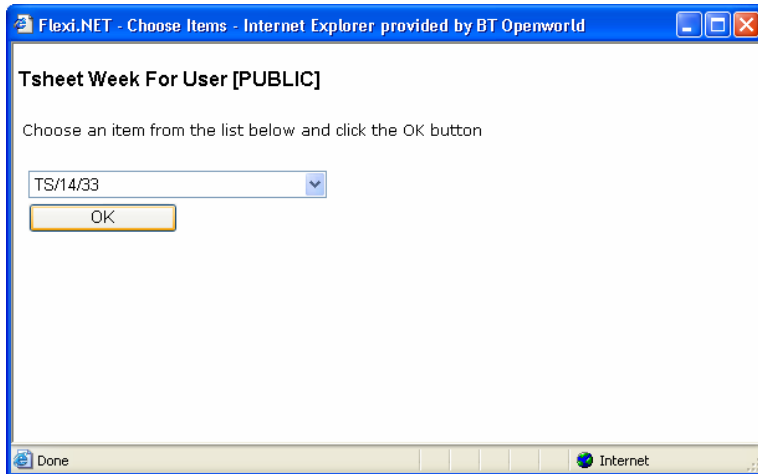
- Command: RunMacro
- Macro: Time Sheet Copy Lines
- Target: POPUP2

The dialog box has a "Set" button and a "Done" button at the bottom. The background interface shows a "Field List" with "Tsheets Week" selected.

That's all there is to it to achieve a new button fully integrated into the database and with all the required functionality behind it.



STEP 4 - Deploying to the users...



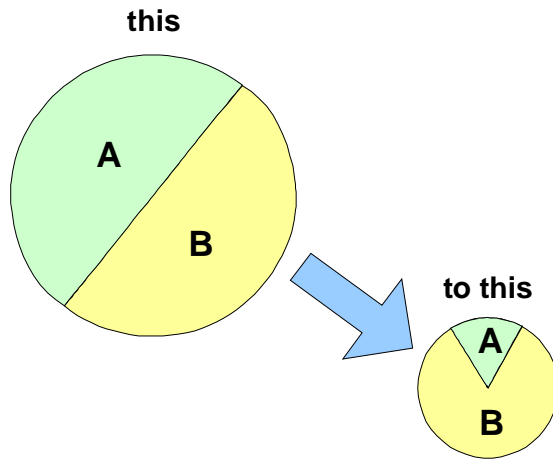
**Done!**

**Summary**

Pyinna offers very significant productivity gains over other conventional methods. These gains are transferred into very large time savings and cost reductions.

All Business Systems require: -	Achieving this using Conventional Development Tools	Achieving this with PYINNA
A. High level process - management programming	Coding using SQL, .NET, C#, VB etc.	<b>Automatic</b>
B. Low level data & number crunching	Coding using SQL, .NET, C#, VB etc.	Coding using SQL, .NET, C#, VB etc.

Use PYINNA to go from..



A business system development using Pyinna is unique, in that it really is *your* system – you have designed it and built it – this means you can change or expand it any time you wish using the same PYINNA development tools. You do this with the confidence that the Pyinna system always maintains the documentation and configuration control of your system. You can develop the system gradually over time, in pace with the initial business demands, user adoption and business evolution.

**Conclusion**

Picture this – you are a business analyst or manager of a department of analysts. You are responsible for delivering IT business systems for your company – solid, reliable, really cost effective solutions, ones that users feel are a benefit to their jobs. You have a stack of requirements on your desk, some are pressing...

You define requirements for each business system, auto-build it using Pyinna and deploy your new system to users in a fraction of the time it would have taken using conventional development methods.

You have just cleared your desk and saved your company a great deal of money.