Estimating with confidence: Applying COSMIC method for estimation in the avionics industry

R. Bridges

European SEPG, 11th – 14th June 2007

helicopters and people you can count on

Introduction

In today’s highly competitive world accurate software estimation can make the difference between successful projects and dismal failures.

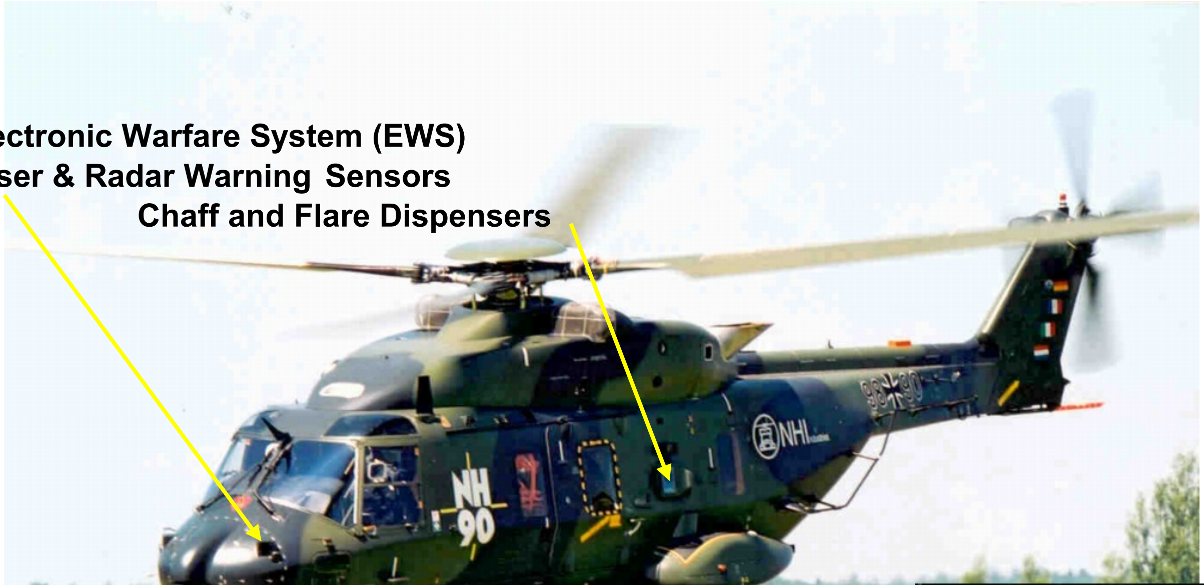
Many estimates are quickly based upon incomplete and poor quality data often using a single experts best guesses.

This presentation shows how the COSMIC Full Function Point method has been applied on a military avionics mission computer. Due to time constraints, automated counting was used wherever possible. The available artifacts from the software development process have been analysed to:

a) Predict the size of an Operational Function (OPF) from System Specifications
b) Determine the productivity of the software development process

The derived data and models have been then used to estimate the size and development effort of new system features based upon System Engineering input from another programme using the “Early COSMIC–FFP” method.

NH90 Tactical Transport Helicopter (TTH) Mission System Components

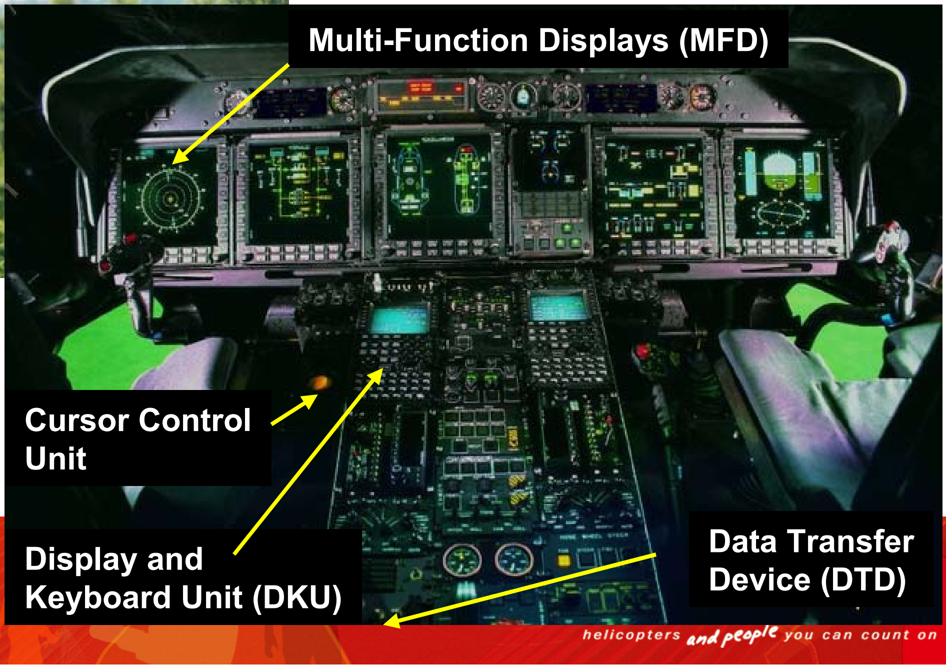


Electronic Warfare System (EWS)
Laser & Radar Warning Sensors
Chaff and Flare Dispensers

Forward Looking Infra Red (FLIR)

Weather Radar (WXR)

Helmet Mounted Sight (HMSD)



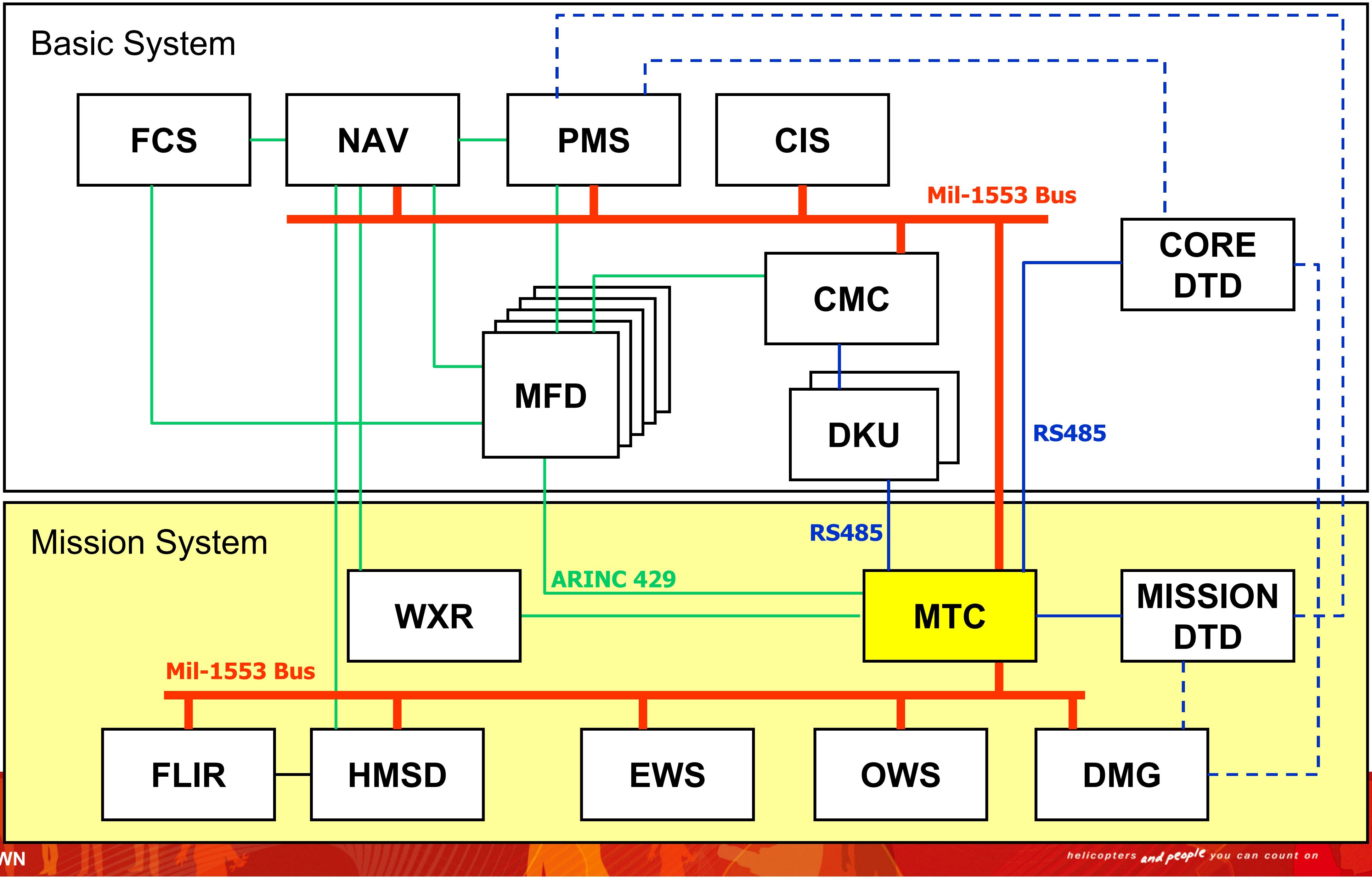
Multi-Function Displays (MFD)

Cursor Control Unit

Display and Keyboard Unit (DKU)

Data Transfer Device (DTD)

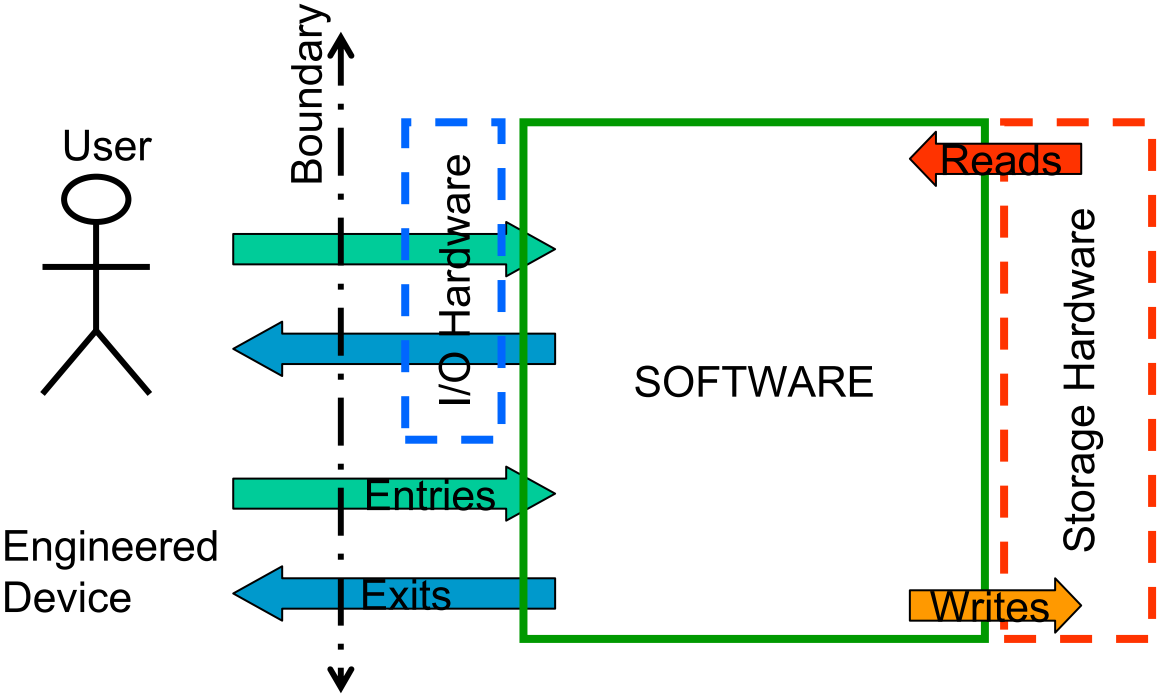
NH90 TTH Avionics System Architecture



COSMIC Full Function Point Method

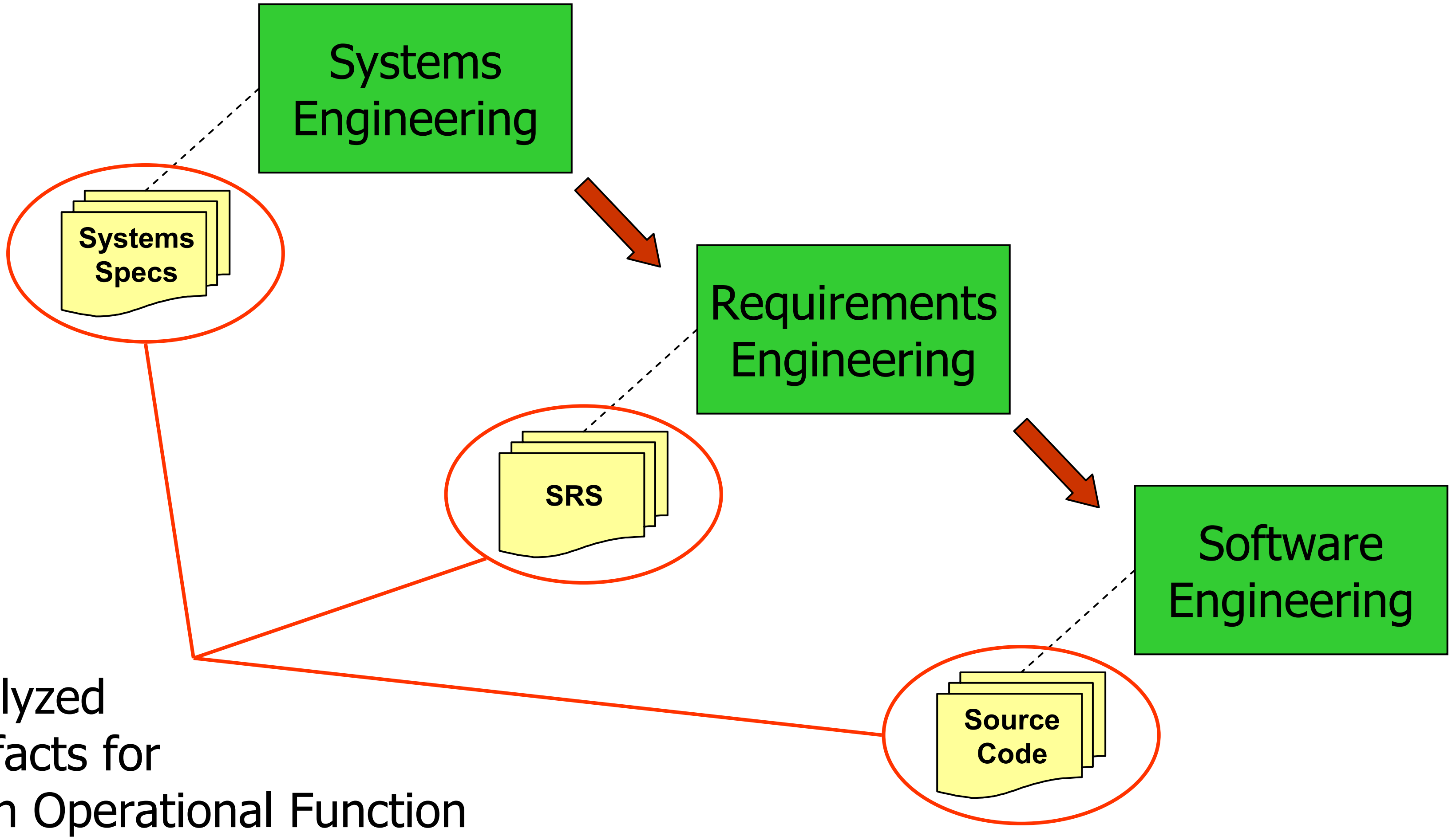
The COSMIC-FFP measurement method involves applying a set of models, rules and procedures to a given piece of software as it is perceived from the perspective of its Functional User Requirements. The result of the application of these models, rules and procedures is a numerical "value of a quantity" representing the functional size of the software, as measured from its Functional User Requirements.

Each instance of a basic data movement, Entry, eXit, Read or Write is counted as one function point. The sum of all basic data movements is determined for the artifact under analysis



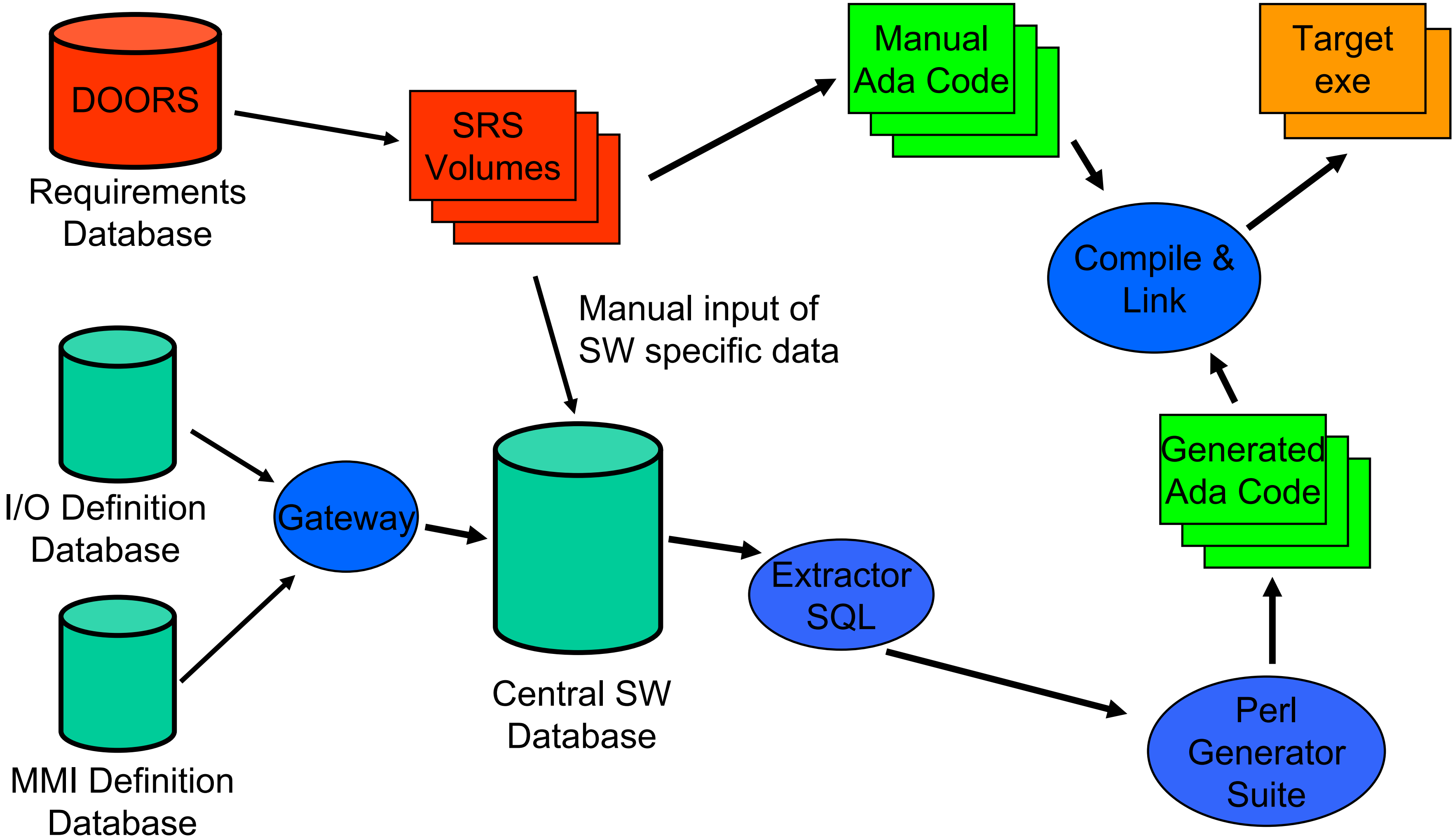
$$\textbf{Size\_Cfsu} = \Sigma\ \text{size(E)} + \Sigma\ \text{size(X)} + \Sigma\ \text{size(R)} + \Sigma\ \text{size(W)}$$

Software Development Model and Artifacts



Analyzed artifacts for each Operational Function

Software Development Process



eurocopter
an EADS Company
helicopters and people you can count on

Example - Function Point Counts for System Specifications

System Specifications called “Functional Chains” show the interaction between subsystems, i.e. process flows, required to provide a “Feature” to a Mission System User.

The size adjusted process flows are counted

- Use actual data flow sizes
- or estimations with normalized values, e.g. (S => 3, M => 10, L => 30, XL => 100)
- The Cfsu for all “Features” are then summed for the system specification

The example shows the forwarding of FLIR Gain and Level to the MFD for display

Functional Chain Identifier	**Source**	**Sink**	**Description**	**Type**	**Est Size**	**Act Size**	**Cfsu**
FC–FLIR_GAIN_LEVEL	FLIR	MTC	Info with gain/level correction	Dataflow		2	
	MTC	MFD	Symbol and value of gain/level correction	Dataflow		2	
Total							4

Software Requirement Specifications (SRS)

Software requirement modules defined in a DOORs database are parsed using DOORS eXtension Language (DXL) scripts . Data flows with names conforming to design guidelines are identified. CSV tables are generated for further processing as simple spreadsheets.

The following data flow classes are identified:

- External-IO => Entry, Exit
- MMI Commands and Parameter Data flows => Entry, Read
- MMI Display Data flows => Exit, Write
- Inter-OPF Events Messages and Data flows => Entry, Read, Write
- Internal OPF Data flows => Read, Write
- Unknown => Requires manual check

Unrecognized data flows may indicate that the "naming rule definitions" need to be extended for new operational functions, or "tuned" for non standard naming conventions, or indicate quality problems in the SRS.

The Cfsu for all requirements are then summed for the SRS.

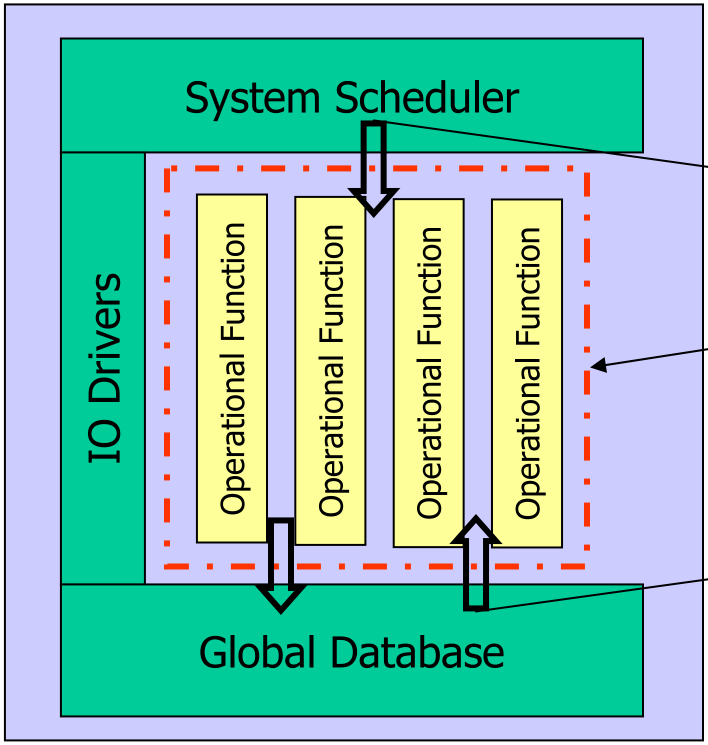
Example - Function Point Counts for SRS

```
Req. TMS-SRS-FLIR-96: FLIR_DATA_RELAY: FLIR Status Data to MFD
The FLIR gain and level settings shall be encapsulated in the
information flow FLIR_MFD and provided to the external activity
CMTC_COMPUTER_MGT for display on the MFD.
/FLIR_MFD
   MTCC_MFD_FLIR_GAIN_CORR:=FLIR_MTCC_GAIN_CORR;
   MTCC_MFD_FLIR_LEVEL_CORR:=FLIR_MTCC_LEVEL_CORR;;
Test Method: Test
End-Req.
```

Requirement Identifier	ENTRY	EXIT	READ	WRITE	Cfsu
Req. TMS-SRS-FLIR-96	2	2	0	0	4

Software Model - Identification of Boundary

- Identification of Software Boundary
- Operational Functions data flows are counted
- IO Drivers and System Functions are not considered



OPF Event Handling Invocation
=> Entry

Identified Boundary

- OPF access to Data Interfaces
  - External => Entry / Exit
  - Inter OPF => Read / Write
  - OPF Internal => Read / Write

Example - Source Code FP Counting

```
Flir_Lld.Get_Get_Flir_Gain_Corr
    (Value => Flir_Gain_Corr);
Flir_Lld.Get_Get_Flir_Level_Corr
    (Value => Flir_Level_Corr);
...........
Mfd_Lld.Put_Mfd_Flir_Gain_Corr
    (Value => Flir_Gain_Corr,
     Index_1 => Mfd_Display);
Mfd_Lld.Put_Mfd_Flir_Level_Corr
    (Value => Flir_Level_Corr,
     Index_1 => Mfd_Display);
```

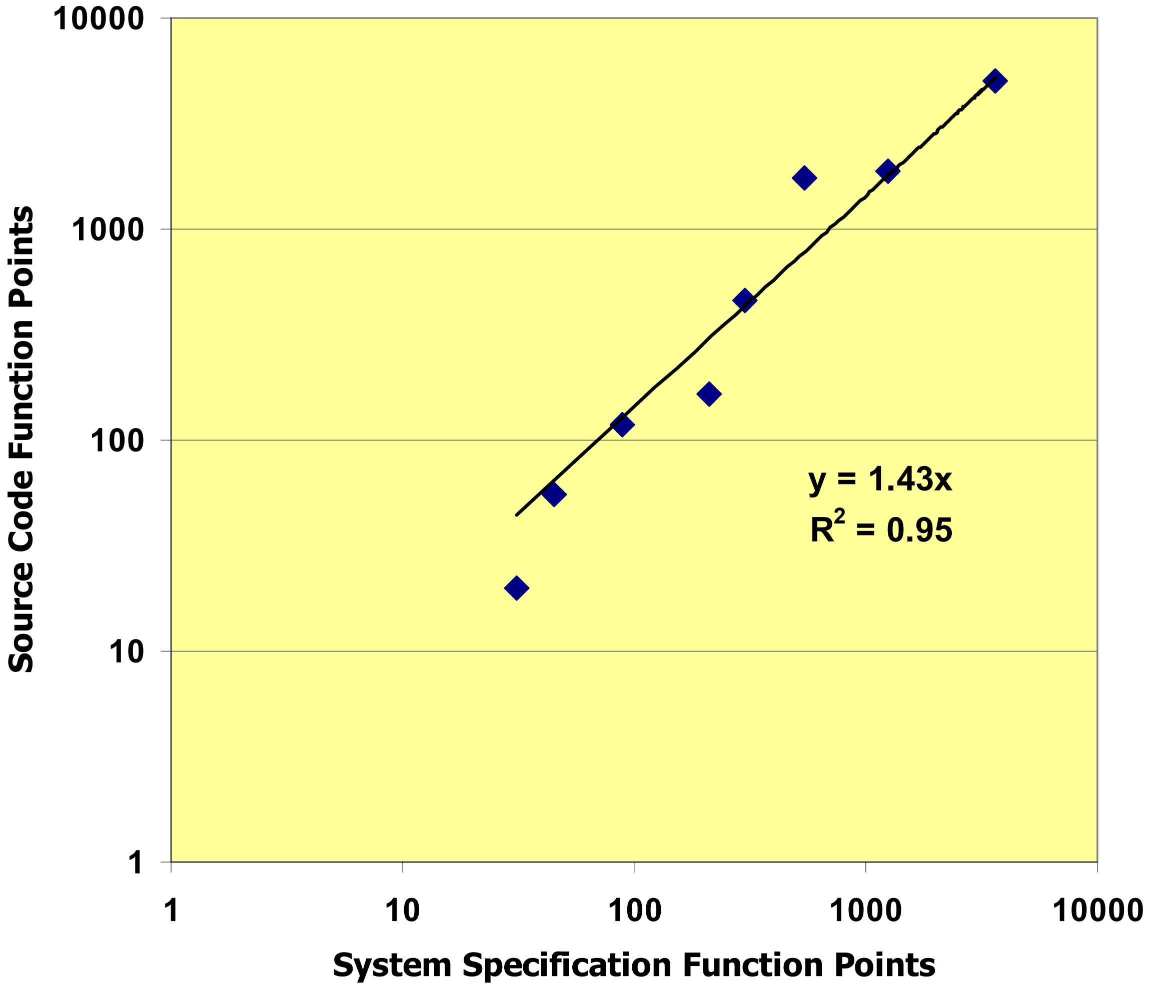
- Scripts are used to count all flows which cross the identified boundary
- All flows follow strict naming convention, this allows automated counting

Computer Software Unit	**ENTRY**	**EXIT**	**READ**	**WRITE**	**Cfsu**
Flir_Opf.Process_Mode_Report	2	2	0	0	4

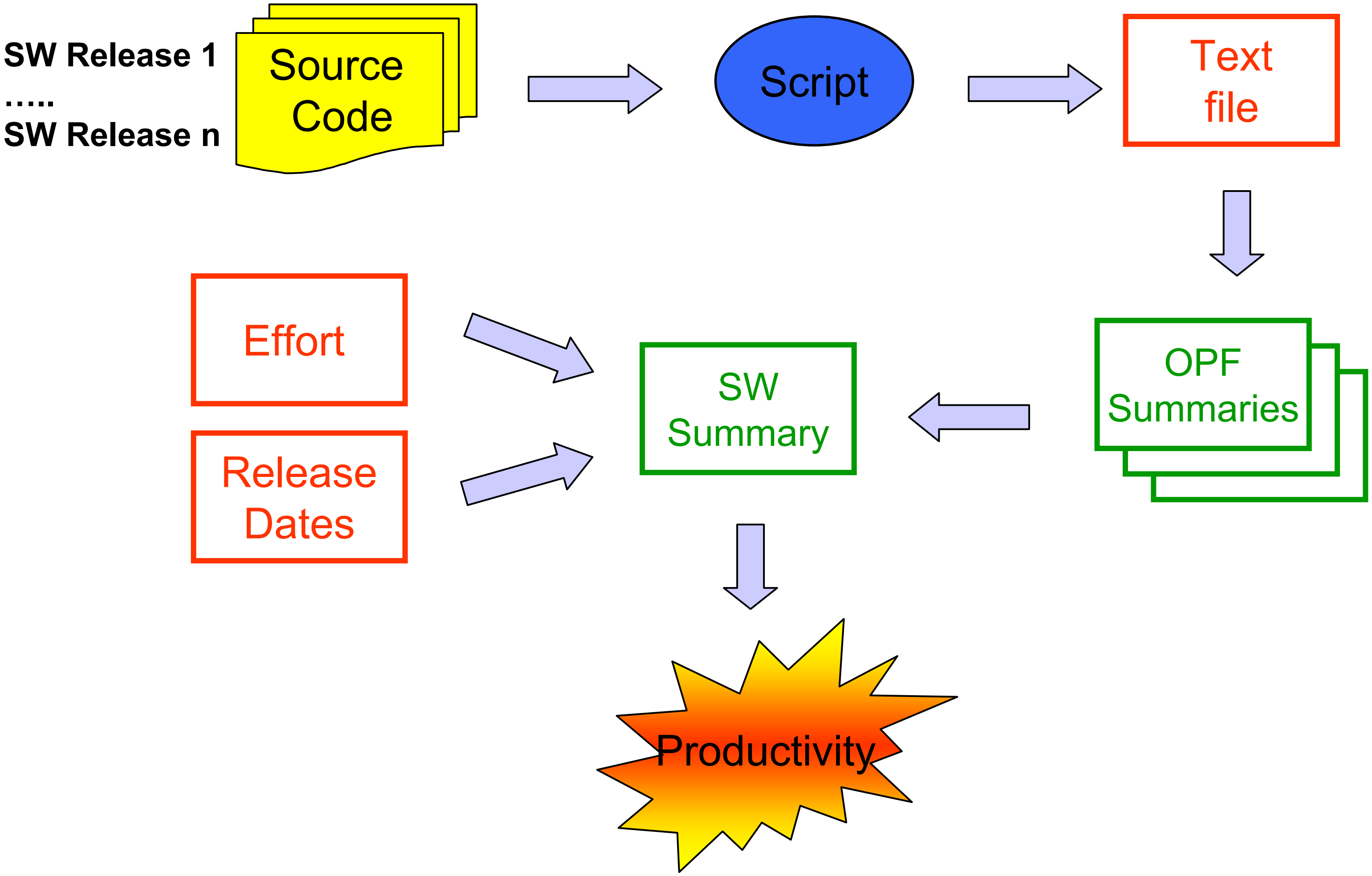
Estimation Model - Basic Data

Operational Function	FC FP	SRS FP	Code FP	SLOC
Cursor Control (CuCo)	45	148	55	910
Digital Map (DMG)	1239	4658	1493	22544
Electronic Warfare System (EWS)	301	849	462	14617
Forward Looking Infra Red (FLIR)	31	51	20	3819
Helmet Mounted Sight / Display (HMSD)	89	236	118	790
Mission Data Transfer Device (MDTD)	540	1344	1751	14617
Mission Flight Plan Management (MM)	3644	7748	5053	52881
Mission System Monitor (MSM)	413	330	154	2610
Weather Radar (WXR)	211	368	165	1132

Estimation Model - System Specification to Source Code



Software Development Productivity - Process



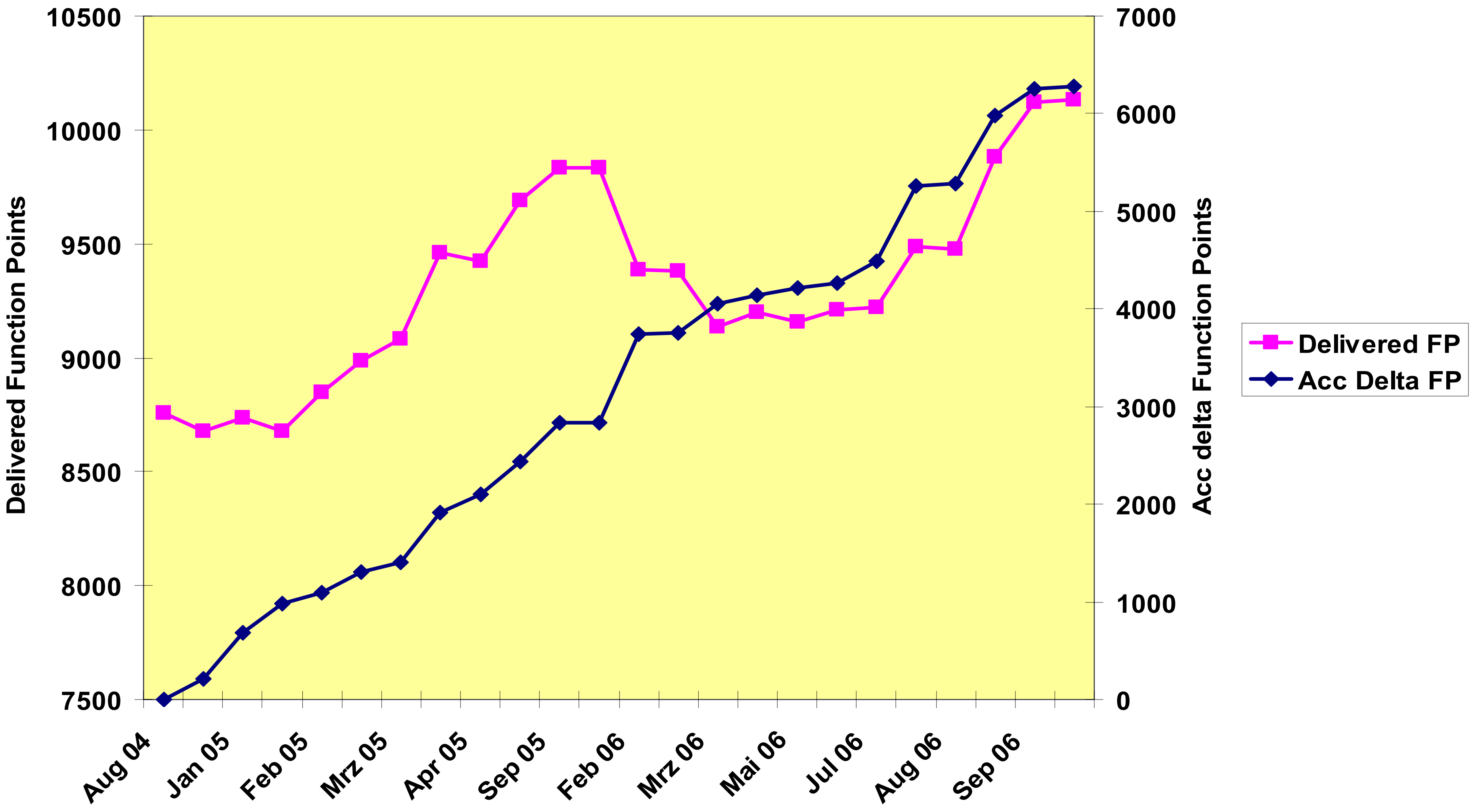
Source Code FP Evolution for the Digital Map OPF

Date	Invocation ENTRY			IO ENTRY		IO EXIT		READ	WRITE	Cfsu	delta Csfu	Acc Delta
Aug. 04	10	18	33	142	1	544	23	302	158	1231	1231	1231
Feb. 05	10	18	57	205	1	367	21	332	196	1207	334	1565
Mrz. 05	10	18	57	210	1	390	25	362	208	1281	74	1639
Apr. 05	10	18	57	255	1	410	31	354	219	1355	90	1729
Apr. 05	10	18	57	256	1	404	31	358	218	1353	12	1741
Jul. 05	10	18	58	269	1	397	42	412	256	1463	124	1865
Sep. 05	10	25	58	269	1	395	42	449	244	1493	58	1923
Nov. 05	10	25	58	269	1	395	42	449	244	1493	0	1923
Feb. 06	10	25	59	264	1	232	22	346	213	1172	323	2246
Jul. 06	12	25	20	255	1	230	24	279	216	1062	124	2370
Aug. 06	12	25	59	255	2	230	24	290	229	1126	64	2434
Sep. 06	12	25	59	253	2	236	23	294	232	1136	16	2450

Source Code FP Evolution for the Complete Software

Software Release	Date	Total FP	Delta FP	Acc Delta FP
mtc_3.1.1.2	Aug. 2004	8757	8757	0
mtc_3.1.2.1	Feb. 2005	8678	988	988
mtc_3.1.2.2	Mrz. 2005	9084	423	1411
mtc_3.1.2.3	Apr. 2005	9459	499	1910
mtc_3.1.2.4	Apr. 2005	9424	195	2105
mtc_3.1.3.5	Jul. 2005	9692	334	2439
mtc_3.1.4.6	Sep. 2005	9834	390	2829
mtc_3.1.5.8.1.1	Feb. 2006	9466	914	3743
mtc_3.1.5.8.1.3	Mrz. 2006	9216	310	4053
mtc_3.1.5.8.1.7	Jun. 2006	9290	210	4263
mtc_3.1.5.9	Jul. 2006	9299	231	4494
mtc_3.1.6.9	Aug. 2006	9573	787	5281
mtc_3.1.6.9.1.4	Sep. 2006	9965	703	5984
mtc_3.1.6.10	Sep. 2006	10204	296	6280

Function Point Evolution - Aug 2004 to Oct 2006



Effort Expended - Aug 2004 to Oct 2006

6280 Functions points were Implemented, Modified or Deleted in period August 2004 to October 2006.

	Requirements Engineering	Software Development	Software Test	Configuration Management	Totals
Hours	10000	50000	10000	2000	72000
Hours / FP	1.59	7.96	1.59	N.A.	11.46

Early COSMIC-FFP method – An Application

The developed models was used to provide an effort estimate to implement a data link based upon System Engineering inputs from another programme. The estimate was performed to the Early COSMIC-FFP method as described in COSMIC Measurement Manual v2.2 Jan 2003

The following input data were used to derive an effort estimation.

- A list of the required data link "features"
- A representative functional size of a "feature" implemented in the "NH90 architecture"
- The system specification to source code estimation model
- Current productivity figures

Transparency of the effort estimation process allows optimisation and "what if" analysis e.g. Trade off number of features and complexity of each feature.

Monitoring of functional size through development gives early warning of "requirements creep" and potential cost overruns.

Conclusions

COSMIC FFP Function Point counts were measured at the following abstractions:

- System Requirements
- Software Requirements
- SW Source Code

An analysis of the source code was performed for a set of releases over a 2 year period. Expended effort was used to determine productivity data.

Use of project enforced dataflow naming conventions allowed automated counting using scripts.

The derived models and productivity data were then used to perform estimations using the “Early COSMIC-FFP” method described in the COSMIC Measurement Manual.

Thank You !