



DIRECTIONS
E M E A

THE HAGUE | THE NETHERLANDS

Infoma ▶

HOW TO MANAGE THE LIFECYCLE OF VOLUME VERTICALS

Tobias Fenster
Directions EMEA
The Hague, 31.10.2018



Tobias Fenster

CTO at Axians Infoma

Microsoft MVP for Business Applications

[†]@tobiasfenster and [†]<https://navblog.axians-infoma.com>

Agenda

- ▶ Setting the stage: **Overview Infoma newsystem and TFS**
- ▶ Part one: **From customer request to bug / user story**
- ▶ Part two: **Release planning and development in TFS**
- ▶ Part three: **Continuous integration and quality assurance**
- ▶ Part four: **Full traceability and release pipelines**

- ▶ Infoma newsystem:
- ▶ Vertical solution for public sector (cities / municipalities / districts in Germany and Austria) and churches: Finance, taxes, facility management, public institutions, document oriented workflow, e-Payment and more
 - Sister company Axians IT&T with a very similar business in Switzerland
- ▶ >1.200 customers on the same code base (from current to about 12 months old release), currently on NAV 2017 moving to Business Central
- ▶ Infoma also offers introduction and implementation, operations support and individualization (in separate teams)
 - Separation of product development and individual customization projects was very important
 - After trying some other setups, bringing together pm, dev and backoffice support for specific application areas in one unit works well
- ▶ Customers download and (automatically) self-install new releases
 - Next step probably will be to offer this as a WebService

- ▶ Microsoft Team Foundation Server (TFS):
- ▶ Supports full **Application Lifecycle Management**: from idea to monitored solution in production
- ▶ We'll cover **plan, code, build, test** and **release** in TFS 2018, most in C/AL or AL based areas
- ▶ **Scrum-based** development methodology
 - Others are possible as well
 - A lot of configurability and extendability including a big marketplace
- ▶ Runs **On-Prem at Infoma** (even deeper extendability), also available as cloud-based SaaS offering **Azure DevOps** (formerly Visual Studio Team Services)
 - Quite similar to Business Central OnPrem vs. Business Central Cloud at the moment

- ▶ One source of "truth" with **all relevant current information** and as few context changes as possible
- ▶ Minimize **manual and/or repetitive tasks** for development, product management and release management
 - Handling a lot of releases per year (every 4-6 weeks) is not reasonably manageable otherwise
 - Automate creation of tasks, test cases, documentation; closing tasks, related issues, ...
- ▶ Maximize **release quality**
 - One of the base pillars of software development: A bug found before delivery to the customer costs **only a fraction** of a bug found at the customer → **fail fast, fail early**
 - **Continuously updated** test environments
 - **Self-service** creation of test environments
 - **Automated testing** is work in progress, e.g. migration to BC OnPrem already profits
- ▶ **Full traceability** from bug report / feature request through code and test to build and release → AL
- ▶ Good balance between **data security** (GDPR) and **as little friction as possible** during the full development process

AGENDA

- ▶ Setting the stage: Overview Infoma newsystem and TFS
- ▶ Part one: From customer request to bug / user story
- ▶ Part two: Release planning and development in TFS
- ▶ Part three: Continuous integration and quality assurance
- ▶ Part four: Full traceability and release pipelines

PART ONE FROM CUSTOMER REQUEST TO BUG / USER STORY

- ▶ Customer enters "ticket": **bug report or request for new feature** (web interface based on IBM Notes)
- ▶ If 1st level support classifies this as actual bug (usually by creating a working repo on an internal or the customers's database → tool for db management) or feature request, a **linked WorkItem (UserStory or Bug)** in TFS is generated
 - When a release is published, all support tickets that were closed are automatically closed as well → earlier leads to **frustration** for the customer ("You tell me it is done, now give me the release")
- ▶ Bugs get **classified and prioritized**, feature requests with only middle to long term chances of being done are shown in **voting platform** somewhat similar to UserVoice
 - Finished proposals are also only closed **when the release is published**
- ▶ Both tickets and proposals for voting are visible directly integrated in TFS → no context switching
 - **Time recording** is integrated as well through a WebService call to our NAV-based ERP
- ▶ **Legal requiremens** and **internally planned new features** are also classified and prioritized
- ▶ Centralized **dashboard for open issues** in various stages with short and long term goals

DEMO PART ONE

AGENDA

- ▶ Setting the stage: Overview Infoma newsystem and TFS
- ▶ Part one: From customer request to bug / user story
- ▶ Part two: Release planning and development in TFS
- ▶ Part three: Continuous integration and quality assurance
- ▶ Part four: Full traceability and release pipelines

PART TWO

RELEASE PLANNING AND DEVELOPMENT IN TFS

- ▶ Different WorkItem paths according to teams size but mainly **inbox – discussion – waiting for concept – planning done – prepared for development – development** (in Scrum sprints with retro, planning etc.)
 - Sometimes for a couple of weeks we adopt a more **Kanban like style** with the goal of fixing **as many issues as possible with the least possible overhead** → works better with more experienced developers
- ▶ Release planning through **prioritization** (backlog grooming) and field "**planned release**", but broad topics in Office as TFS didn't work so well for us in that area
 - We are not using Epics and Features as mainly intended in TFS...
- ▶ Development consists of "doing", "crosstest and code review" and "transfer to release db"
 - Very much looking forward to (Git or TFVC) **pull requests** for our main solution
 - Small tool for **transferring C/AL objects** to the release database with two- or three-way diff
- ▶ PM tests as much as possible **during sprints** → how much is possible depends on the overall workload

DEMO PART TWO

AGENDA

- ▶ Setting the stage: Overview Infoma newsystem and TFS
- ▶ Part one: From customer request to bug / user story
- ▶ Part two: Release planning and development in TFS
- ▶ Part three: Continuous integration and quality assurance
- ▶ Part four: Full traceability and release pipelines

- ▶ Every night **automated build of a full release** – same as for actual release
 - Again: **fail fast, fail early**
 - No "related changes" yet as we are not using integrated source control for our main solution, but for our AL extension
- ▶ **Automated deployment of changed objects** to test databases (no full install yet) and **auto-tests**
 - Enables product managers to **test with good data** (even some customer databases who agreed to let us test with their data) while development is still working → if possible, **don't wait for release tests**
- ▶ Test cases for **every** closed WorkItem
 - Make sure you **test everything at least once**, no matter how small
- ▶ **Automatically generated PDF documentation** from content in WorkItems
 - Release notes for major releases usually with up to 150 pages or more, which gets ugly in Word
- ▶ After release: Docker-based **self-service version environments** for "did this work in release X and break in release Y?" tests including the correct NAV CUs
- ▶ **Beta workshops** with selected customers and partners together with product managers

DEMO PART THREE

AGENDA

- ▶ Setting the stage: Overview Infoma newsystem and TFS
- ▶ Part one: From customer request to bug / user story
- ▶ Part two: Release planning and development in TFS
- ▶ Part three: Continuous integration and quality assurance
- ▶ Part four: Full traceability and release pipelines

PART FOUR FULL TRACEABILITY AND RELEASE PIPELINES

- ▶ Small **AL Extension** (Entry Search and Find with phonetic search) published in AppSource
- ▶ **Code and WorkItems** in TFS
- ▶ Setup of Team Foundation Version Control as alternative backend for source control in Visual Studio Code
- ▶ **Automated build and test** on every code change and nightly
- ▶ No publish as we don't run customer environments
 - Will show you the **release pipeline** for one of our internal tools in TFS

DEMO PART FOUR





ANY QUESTIONS I CAN TRY TO ANSWER?

[↑]@tobiasfenster

[↑]<https://navblog.axians-infoma.de>

[↑]tobias.fenster@axians-infoma.de

