WP6/D1
Requirements Analysis

Technical Report

Contributors:

Yves Ledru	LIG	Author
Yoann Blein	LIG	Author
Lydie du Bousquet	LIG	Author
Roland Groz	LIG	Author
Arnaud Clère	MinMaxMedical	Author
Fabrice Bertrand	Blue Ortho	Author

Partners:

Partly funded by:

Contents

1. Introduction

This document presents the TKA (Total Knee Arthroplasty) product and the requirements that the MODMED Domain Specific Language (DSL) should support. The DSL expresses properties of the traces. These requirements are taken into account for the design of the DSL, which takes place in Work Package WP1 of the MODMED project.

Definitions

Requirements*: Requirement on functionality, performance, safety, etc. that Blue Ortho product must fulfil and that is checked during the Verification and Validation procedure and the Post-Market Surveillance procedure*
Specification*: Technical description at various levels of details of how the product fulfils the requirement.*
Traces*: Log of events by software and traces of software execution with associated data providing a partial view of what happened during the aided medical intervention*
Properties*: Boolean functions of traces; a property is true for a given trace if the trace satisfies the constraint on values and events expressed by this property. Properties can be used to express a requirement to be checked, or to extract subsets of relevant traces to investigate a specific issue.*

2. Overall Requirements Engineering Process

The MODMED project involves three partners:

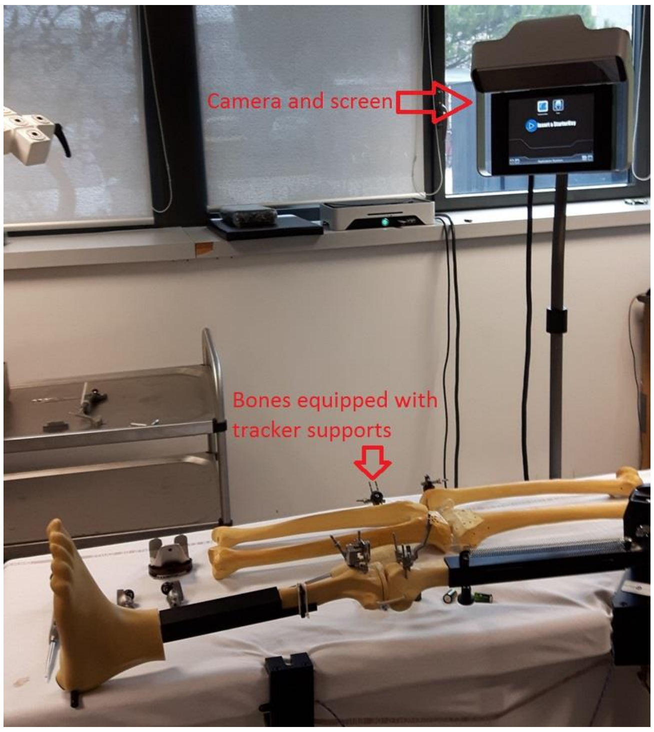
- MinMaxMedical (MMM), a software components and services provider,
- Blue Ortho (BO), which develops medical devices such as the TKA product,
- Laboratoire d’Informatique de Grenoble (LIG), a research laboratory.

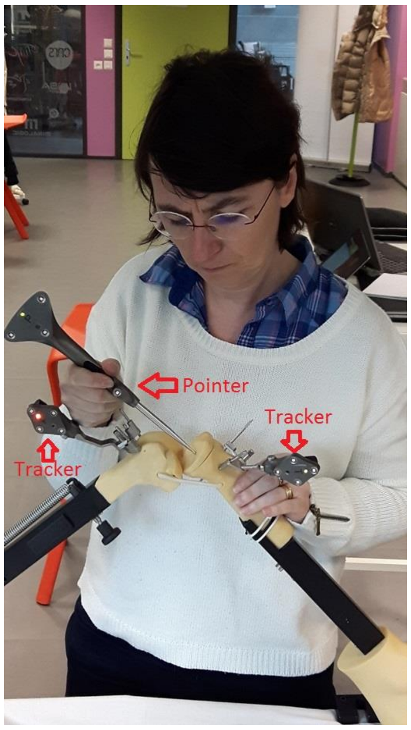
The TKA product has been chosen as case study for this project because, on the one hand, it appears as representative of Medical Cyber-Physical Systems (MCPS), and on the other hand, it has been used worldwide for several years and has acquired more than 7000 traces of execution. These traces provide a rich raw material for the DSL of the MODMED project.

The requirements engineering performed in this case study had two goals: first, the partners had to understand the TKA product, and the contents of its traces; second, they had to identify a set of properties that the DSL would express and evaluate on the traces.

To that end, the partners held about 15 meetings (2-3 hours each) between October 2015 and May 2016. The participants to these meetings were Arnaud Clère and Vivien Delmon (MMM), Fabrice Bertrand (BO), Yoann Blein, Lydie du Bousquet, Roland Groz and Yves Ledru (LIG). The first meetings were dedicated to the presentation of the TKA product. After a PowerPoint presentation, a hands-on session was scheduled to provide the project members with a practical understanding of the product.

The following images provide a view of the TKA product (on the left) and of the hands-on session (on the right). The system includes a set of trackers, firmly attached to the bones of the patient, whose position and orientation can be detected by a 3D camera. The camera also detects a pointer device, which is used to acquire the position of some anatomical points. The surgeon interacts with the product through a touch screen, and using the pointer device. When the system has acquired all anatomic information, the surgeon can plan several cuts in order to place the prosthesis. The system will then help him to position cutting guides.





Every acquisition and interaction is logged by the TKA product into a large trace. This trace is initially designed to store sufficient information to understand what went on during a surgery. The sequencing of operations appearing on the trace is ruled by a configurable state machine, tailored to the surgeon's planning of operations.

After this introduction to the TKA product, a significant time was dedicated to a detailed understanding of the traces, and associated state machines. Two traces of actual surgeries carefully selected by BO as representative were reviewed in detail, and a few others were provided to illustrate particular requirements. The goal was then to identify representative properties that can be evaluated on the traces. In order to identify these properties, the members of the project got access to confidential documents: the "FUNctional SPecification" document of TKA, which describes the specifications of the whole system, and the "TEChnical SPecification" focusing on the TKA software.

3. Context

Description of TKA and its use

TKA is an application to guide Total Knee Arthroplasty operations, i.e. the replacement of both tibial and femoral cartilages with implants, using the ExactechGPS MCPS. TKA is composed of software, mechanical and electronical components described below:

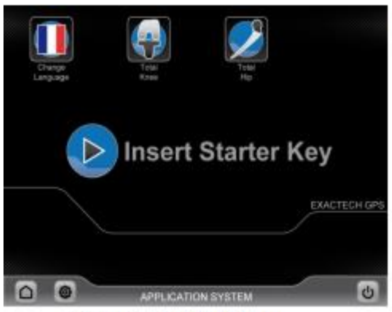
Pico Trackers

Total Knee Instruments

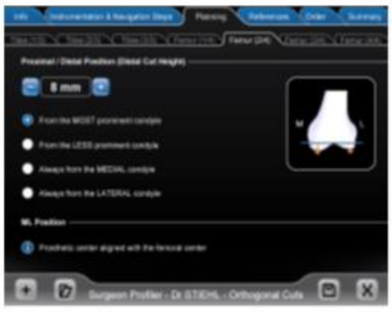
Starter Key

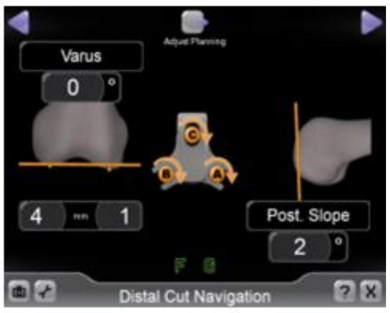
Disposable Batteries

Disposable Drape



Application System


Info
Instrumentation & Navigation Steps
Planning
References
Order
Summary
8 mm
Surgeon Profiler - Dr STIEHL - Orthogonal Cuts
Surgeon Profiler

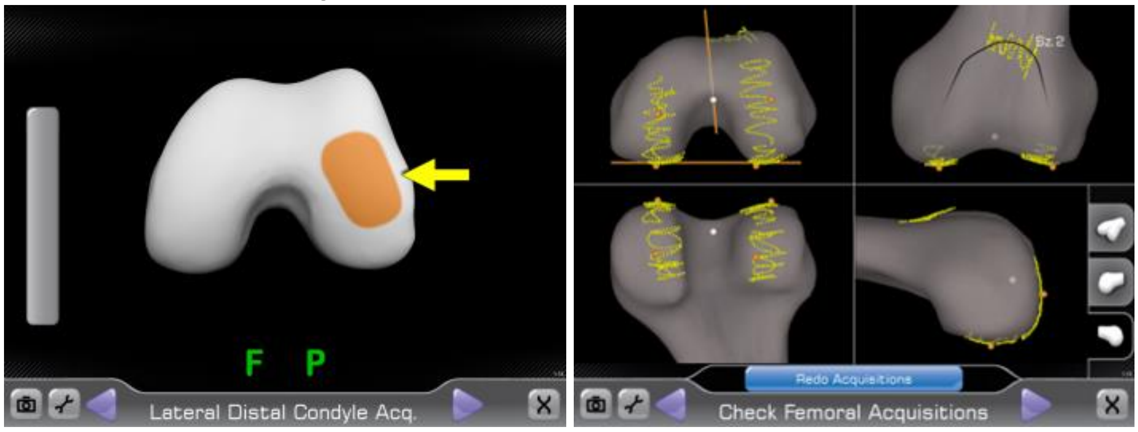


Total Knee Application

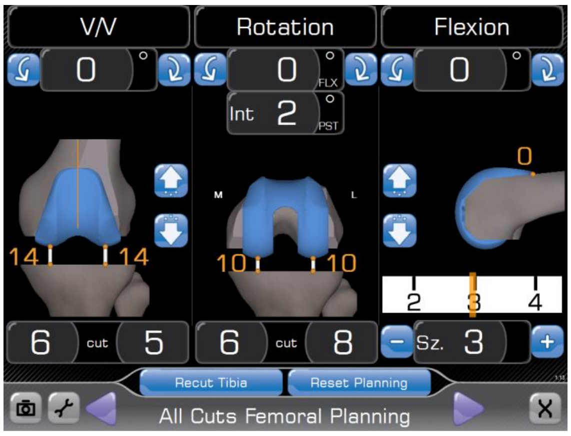
The whole system can be manipulated directly by the surgeon or his assistant:

Using TKA mainly consists in the following sequence of operations:

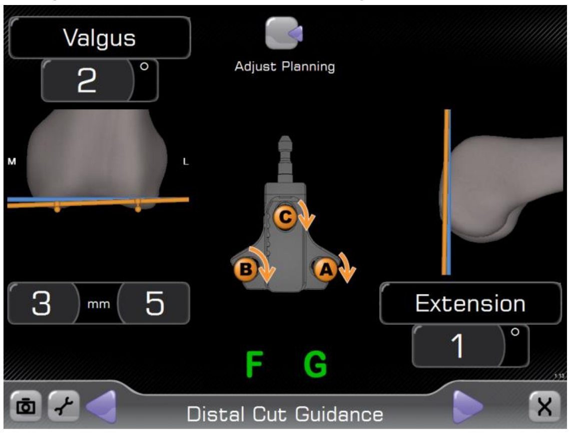
1. **Digitization** (acquiring a digital model of patient's anatomy)
Example of GUI steps to digitize and check the patient's femur dimensions:



2. **Decision** (adjusting the preoperative planning of implant sizes and position)
Example of per-operative selection and positioning of a particular implant:



3. **Action** (positioning tracked cutting guides to cut bones)
Example of cutting guides adjustment on the patient to realize the planned implant positioning (each implant requires several bone cuts):



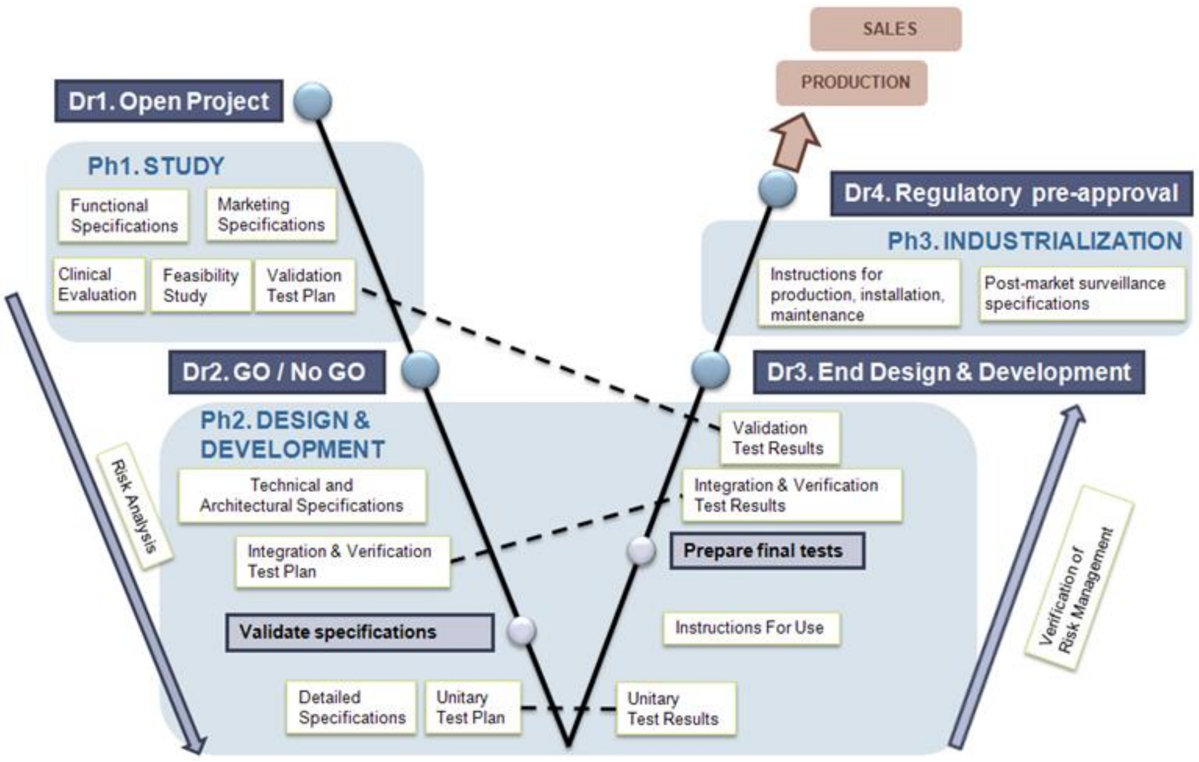
Blue Ortho (BO) Procedures

Study, Design and Development (D&D)

The development of MCPS at BO follows a classical V-Model methodology complying with ISO 13485 complemented with:

1. An end-to-end requirements management complying with ISO 62304 standard and
2. A risk analysis of the medical device complying with ISO 14971 standards.

The BO V-Model can be split in 3 levels depicted by dashed lines in the following diagram extracted from BO Quality Management System:


SALES
PRODUCTION
Dr1. Open Project
Ph1. STUDY
Functional Specifications
Marketing Specifications
Clinical Evaluation
Feasibility Study
Validation Test Plan
Dr4. Regulatory pre-approval
Ph3. INDUSTRIALIZATION
Instructions for production, installation, maintenance
Post-market surveillance specifications
Dr2. GO / No GO
Dr3. End Design & Development
Ph2. DESIGN & DEVELOPMENT
Technical and Architectural Specifications
Integration & Verification Test Plan
Validate specifications
Detailed Specifications
Unitary Test Plan
Validation Test Results
Integration & Verification Test Results
Prepare final tests
Instructions For Use
Unitary Test Results
Risk Analysis
Verification of Risk Management

From the least to the most detailed, we can name these 3 levels: User, Technical and Implementation.

User level: from Intended Use to Validation

At the top level, the Sales, D&D and Quality Managers along with some end-users (surgeons) write a document called “MARketing SPecifications” (MARSP) document describing the intended use of the MCPS and the corresponding high-level user needs, along with a “FUNctional SPecifications” (FUNSP) document describing the MCPS components and high-level usability requirements. At the very end of the D&D procedure, the “Validation” of the MCPS will depend on the result of “Validation tests” typically performed by surgeons on cadavers using a successfully verified MCPS.

MARSP

The goal of this document is to detail MARketing SPecifications of the product by detailing its:

- *Market segment*
- *Business model*
- *Patents*
- *Competitors*
- *Cost and prices targets*
- *Planning*

FUNSP
The goal of this document is to detail FUNctional SPecifications of the product by detailing its:

- *Description and medical device classification*
- *Functions, constraints and validation test plan*
- *Architecture and verification test plan*
- *Product risk management and risk measures test plan*
- *Packaging test plan*

Technical level: from Design to Verification

At the middle level, the D&D and Quality Managers, with the help of their Engineering teams, write a set of “TEChnical SPecifications” (TECSP) documents which detail these requirements in test cases. The “Verification” of the MCPS consists in performing all these test cases manually and analysing their results. Performing all verification tests, as is required for major revisions of TKA, with both ExactechGPS Hardware (HW) configurations takes approximately 1 man.month.

TECSP
The goal of this document is to describe the TEChnical SPecifications by detailing its:

- *Technical functions*
- *Verification test plan*
- *Architectural design*
- *Interfaces*
- *Integration test plan*
- *Risk measures test plan*
- *Technical traceability*

It is more frequently named by other MCPS manufacturers “Software Requirements Specification” (SRS) to follow the American FDA terminology.

For TKA, all the Software (SW) is described in one TECSP containing 32 requirements named “System Functions”:

F1 – Edit Patient Information
F2 – Edit a profile
F3 – Load a profile
F4 – Connect Camera and Tracker
F5 – Calibrate the pointer tip
F6 – Compute calibration data for the GPS component
F7 – Acquire single point with the pointer
F8 – Acquire cloud of points with the pointer
F9 – Compute Ankle Center reference
F10 – Build Tibia mechanical referential
F11 – Compute Tibia Warping
F12 – Compute Hip Center
F13 – Compute Femur Warping
F14 – Build Femur mechanical referential
F15 – Acquire ligament balancing in flexion
F16 – Plane the tibial cut
F17 – Navigate the tibial cut
F18 – Digitize the tibial cut
F19 – Plan the femoral distal cut
F20 – Navigate the femoral distal cut
F21 – Digitize the femoral distal cut
F22 – Plan the 4-in-1 femoral cut
F23 – Navigate the 4-1 femoral cut
F24 – Create an operative Report
F25 – Control HKA (Hip-Knee-Ankle angle)
F26 – User Interaction
F27 – Anterior Cortex Acquisition
F28 – Plan the 5-in-1 femoral cut
F29 – Acquire ligament balancing in extension
F30 – Redo acquisitions
F31 – Navigate the femoral 5in1 Cuts
F32 – Check if planned cuts are reachable

These requirements are too high level to be formalized as properties of traces. However, it is the level used by Engineers to trace which component is responsible for which “requirement”. This traceability will be used, for instance, to select which verification tests should be performed after a change in a requirement or a component.

These 32 functions are further decomposed by the TECSP in a total of 319 test cases totalling 125 pages. The test cases are written following a few guidelines:

- Avoid duplication in the formulation of “system function” / test case “description” / test case “expected result”
- Prefer a direct formulation of requirements (“what is required” as opposed to “what should/must be avoided”)

For example, “F1 – Edit Patient Information” is decomposed in 4 test cases:

Test N°	Description of the test (including reason(s) for carrying out this test)	Expected results (including how to check results and results acceptance criteria)
	Criteria : Completeness	
F1-T1	On station v1 and station v2 : Check visually the characteristics of the Patient Page window.	Following field are available - Name - First Name - Patient Identification - Date of birth - A virtual keyboard is available
	Criteria : Functional	
F1-T2	Fill the forms as following, use touchscreen to navigate next/previous field Name = <BlueOrtho> FirstName=<> Patient Id = <1234/ABCD/EFGH> Date of birth = <*date of the day*>	The VirtualKeyboard allow to fill the form as expected. It is possible to navigate between fields by using Up/Down arrows It is possible to correct typographic error.
F1-T3	A starter key is plugged. Fill the forms with data. Goto end of workflow and save the OperativeReport.	The data entered in the form are recorded inside the OperativeReport saved on the starter key. There is no patient information recorded in other OperativeReport.
	Criteria : Robustness	
F1-T4	Perform unpredicted actions using - special keyboard combination - Touchscreen - Insertion / removal Key	The behaviour of the software is conforming to performed actions.

While “F17 – Navigate the tibial cut” consist in 10 tests among which:

Test N°	Description of the test (including reason(s) for carrying out this test)	Expected results (including how to check results and results acceptance criteria)
	Criteria : Accuracy	
F17-T2	Initial environment: Profile P1 Action Perform Tibia/Femur acquisitions as expected. (Tibia Center = TestBench point m8) Variables Instrum = GPS Block / External Fixation + Blade Guide Side = Left / Right Perform 10 measurements through the volume of visibility.	see nested table below Expected Standard Deviation < 1.0

Expected Value (+/- 1)	L Cut Height (mm)	M Cut Height (mm)	Varus (°)	Slope (°)
	19.52	9.44	0.12	0.48

Expected Standard Deviation < 1.0

On average, there are 10 tests per function. Only 5 functions have more than 12 tests, and the most complex system function “F28 – Plan the 5in1 femoral cut” has 46 tests. The effort necessary to test each case may greatly vary, depending on the number of initial environments.

These tests focus on the system accuracy and the user workflow. Accuracy cannot be tested with the MCPS alone and is specifically tested with ad-hoc HW test benches. Other accuracy tests are performed during Production procedures described below. Most system functions include a test case like F1-T4 where the tester is free to take unspecified user actions and to appreciate whether the resulting system behaviour is normal. This illustrates the compromise between specifying tests with well-defined acceptance criteria (even informally) and the desire to verify the MCPS under more situations without having the time to precisely describe their input and output.

Verification test cases are the most important source of properties of traces studied by MODMED project.

Implementation level: from Development to Unit tests

At the lowest level, Engineers write “DETailed SPecifications” (DETSP) documents for each component in the form of unitary test plans. They implement the components and write unit tests for each component’s function including tests using lower-level components (SW integration tests). Since all requirements in the DETSP are implemented as unit tests, MODMED partners studied the unit tests rather than the DETSP.

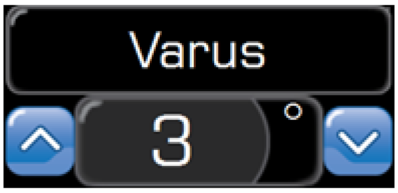
DETSP

The goal of this document is to detail:

- *The detailed specifications and*
- *Test cases and tests carried out during the tests phase of each component.*

TKA SW contains 250 C++ classes which source code also contains the unit tests. The ratio “tests size over implementation size” can reach up to 3 depending on components.

These tests are compiled only in a specific “build configuration” which is used at various times during the development to automatically run more than 600 unit tests. They are written using the QTest library which allows writing unit tests up to GUI components like the VVLabel GUI component which is responsible for accurately displaying named numerical values with the appropriate unit in various languages.



Example of VVLabel GUI component unit test:

```
void test_01()
{
        QTESTDESC("test VVLabel in 'PLANNING' mode, with 'increase' button
        on left side, and 3degrees of varus.\ncheck title and value texts");
        VVLabel* label = new VVLabel(EditOneValueLabel::INCLEFT_DECRIGHT);
        sendResizeEvents(label);
        label->setMode(PLANNING);
        VarusValgus vv(3 * astk::geom::DEG2RAD, VarusValgus::VARUS);
        label->setValue(vv);
        test::grabWidget(label, "GUI/NavLabels/VVLabel_3VAR_PLNG_incLeft");
        QCOMPARE(label->m_title->text(), QString(tr("Varus")));
        QCOMPARE(label->m_value->text(), QString("3"));
}
```

The unit test results are collected and verified manually before performing the MCPS “Verification”. Being able to run these tests and obtain repeatable test results is of immense value to BO and allowed to manage the proliferation of user options (~30 pages, ~60 options), and translations (English, French, Spanish, German, Italian, and later Japanese, Korean, etc.).

However, the limit of testing is well understood by BO that takes test results as no more than “one evidence that the SW performed well at least once”.

Consequently, BO puts a lot of efforts in SW design to improve the SW safety, like tracking the coordinate systems of geometrical primitives (anatomical axis and planes, physical instrument points, etc.) using dedicated classes (“LocalizedPoint”, “LocalizedPlane”, etc.).

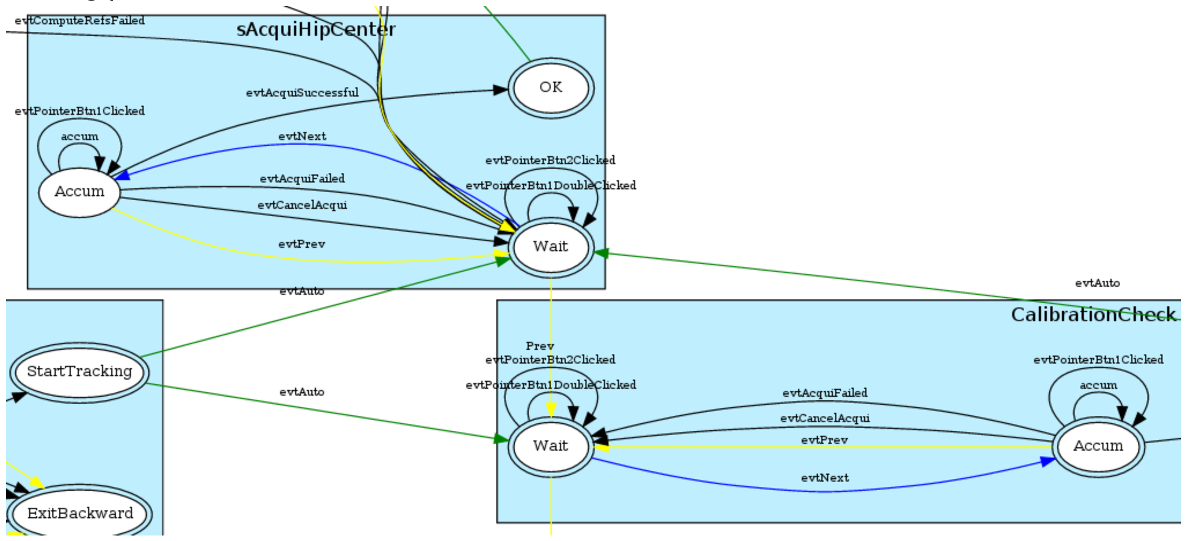
Hierarchical Workflow State Machine

A notable example of BO emphasis on design resulted in putting the responsibility to adapt TKA workflow to the surgeon’s one into a hierarchical state machine which models this workflow and drives the whole UI (from physical tools to GUI widgets). States of this state machine are named “Computer-Aided Surgery Protocol” (CASP). This state machine is dynamically built from pre-tested CASP following a “profile” established by an Exactech representative during an interview of the end-user surgeon. For instance, the surgeon may choose to:

- acquire optional anatomical points like the “Whiteside’s line” (a line following the deepest part of the femoral trochlear groove)

- start bone cuts with the tibia or femur
- etc.

The shape of the state machine (states tree and transitions) at the beginning of TKA is traced in a .dot file that has been thoroughly studied by MODMED partners as it was envisioned as an important but difficult source of properties of traces. Below is a zoom showing parent and child states of this state machine:



Another important thing that BO put in the product to account for the limits of in-house testing was the production of traces by TKA product during all executions, be it for testing or during real surgeries.

Technical log

The first trace that was added is a technical log written as a text file where most trace points will write a single line (but a few trace points may write several lines). Here is an excerpt of such technical log with especially interesting information underlined:

```
MSG 2015.11.11-02:18:01.145 | OPEN
MSG 2015.11.11-02:18:01.145 | Version : 1.15.3
...
MSG 2015.11.11-02:18:01.395 | [EventHandler::performStateEntry] Entering state : mainCasp
MSG 2015.11.11-02:18:01.395 | [EventHandler::performStateEntry] Entering state : mainCasp.Welcome
...
MSG 2015.11.11-02:18:02.659 | [EventHandler::performStateExit] Exiting state : mainCasp.Welcome
MSG 2015.11.11-02:18:02.846 | [EventHandler::performStateEntry] Entering state : mainCasp.Enter Patient Info
MSG 2015.11.11-02:18:11.207 | [BlueApp] Click on Btn Left at pos = (475,95)
...
MSG 2015.11.11-02:18:37.462 | [MainBlueWidget] Screen Btn Next clicked
...
MSG 2015.11.11-02:19:02.410 | [Profile::loadFromXML] file 'C:/.../Dr. XXX - ALL CUTS w ACB.bprofile' loaded successfully
...
MSG 2015.11.11-02:19:07.340 | [FoxDriver::connect] Device 166ec0dd01 connected
...
MSG 2015.11.11-02:19:43.859 | Marker Detected : 16aa1a9401, P001002
...
MSG 2015.11.11-02:20:37.913 | [EventHandler::performStateExit] Exiting state : mainCasp.CalibrationCheck.Wait
MSG 2015.11.11-02:20:37.913 | [EventHandler::performStateEntry] Entering state : mainCasp.CalibrationCheck.Accum
MSG 2015.11.11-02:20:37.929 | [ 0] 0.00416570310465 22519 64.562875356938733 11.860463388262414
…
```

We can see that each “event” is time-stamped and that the aforementioned state machine transitions are traced. A particular attention was paid to trace GUI and sensors events because BO wanted to be able to understand how the MCPS would be used in the market.

Surgery report

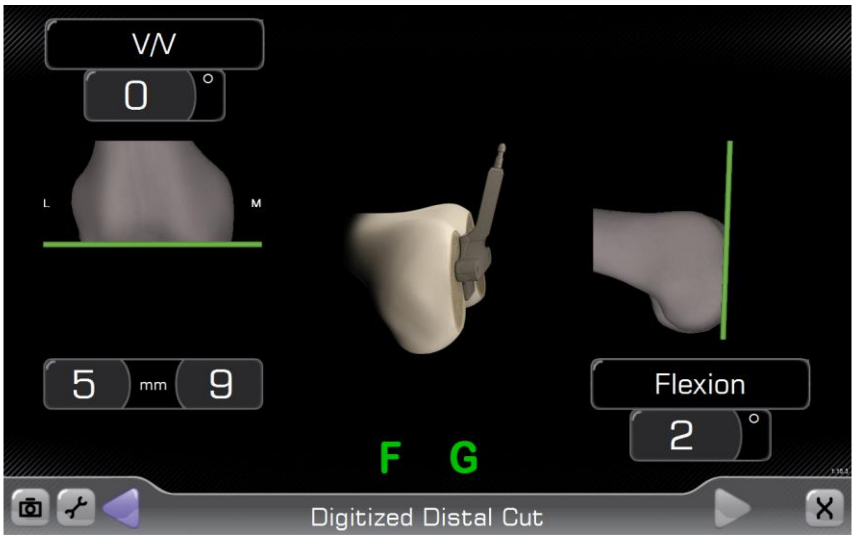
The technical log was quickly complemented with another trace specifically designed to produce various reports on real surgeries. These reports are written by specific trace points in TKA software which produce an XML file allowing various transformations such as the production of HTML reports, SQL INSERT statements, etc. Since the XML format allowed it, the XML trace points were added to the state machine to map the hierarchical states entry/exit events to XML opening/closing <*step*> tags.

However, such structure is not used in the various reports and the fact that it is tied to the state machine hierarchy makes some uses of the trace more complex. Indeed, some MODMED experiments showed that the lack of structure of technical logs is often compensated by the fact that it is more complete than the XML trace. Here is an excerpt of the surgery report corresponding to the above technical log:

```xml
<surgeryReport
  appName="Blue-Ortho-Knee" version="1.15.3" revision="1415"
  surgeon="XXX" launchMode="starterKey"
  time="02:18:01.145" date="2015-11-11">
  <station>
    <boot time="00:34:20.388" date="2015-11-11"/>
    <watchdog version="1.0"/>
    <serialNumber SN="08000214"/>
…
  <step name="mainCasp" time="02:18:01.395">
    <step name="Welcome" time="02:18:01.395">
      <screenshot title="Welcome" id="1" file="./screenshots/screen001.png" time="02:18:02.846"/>
    </step>
…
    <step name="sAcquiHipCenter" time="02:20:38.911">
      <step name="Wait" time="02:20:38.941">
…
      </step>
      <step name="Accum" time="03:06:05.215">
        <logFile>./TechnicalData/hipCenterData1.xml</logFile>
        <lastAcqui>
          <trackerPos elecSN="16ab376f01" … name="F" isVisible="true">
            <localizedTransform valid="true" ref="FoxLocalizer_Ref" name="transfoToTop">
              <transform>-0.90047522500766686 … 750.98583309934361 0 0 0 1</transform>
            </localizedTransform>
            <rms>0.04</rms>
            <distance>792.59</distance>
          </trackerPos>
        </lastAcqui>
        <amplitude>75.20</amplitude>
        <circularity>0.90</circularity>
      </step>
…
```

Other trace data

These two traces are complemented with technical data such as screenshots taken at each step of the surgery. These screenshots contain interesting information related to requirements. The following figure is an example of such screenshot:



However, the traces contain little information about the internal values of the program (variables) and its control flow (method calls). Also, raw data acquired by the sensors is completely absent. Enriching the traces significantly is one of the objectives of the MODMED project.

All in all, real surgeries provide between 5 and 30 MB of trace data, half of which corresponds to screenshots.

Parallel Quality Assurance activities

Quality Engineers perform activities parallel to D&D activities like:

- Approving the traceability of requirements (“System Functions” and “Constraints”) from “user” to “technical” levels
- Conducting a risk analysis with all Engineering teams.

As a result of these QA activities, BO is confident that all TKA requirements are taken into account by TECSP test cases and verified. This comforts MODMED partners in using TECSP as the primary source of requirements to establish properties of traces.

Production

Each MCPS produced according to the validated design is tested again to verify that it fulfills the requirements that could not be ensured by design, especially accuracy requirements. For instance, ad-hoc test benches will be used to put the MCPS in some well-defined state like a sequence of positions and orientations of a tracker in front of the ExactechGPS camera and the accuracy of measures given by the MCPS will be verified.

As in verification activities, information external to the MCPS like the start and end of a test case performed by a tester, or the position of a tracker ensured by the ad-hoc test bench could be merged with the MCPS trace so as to verify accuracy requirements as properties of these merged traces.

Post-Market Surveillance (PMS)

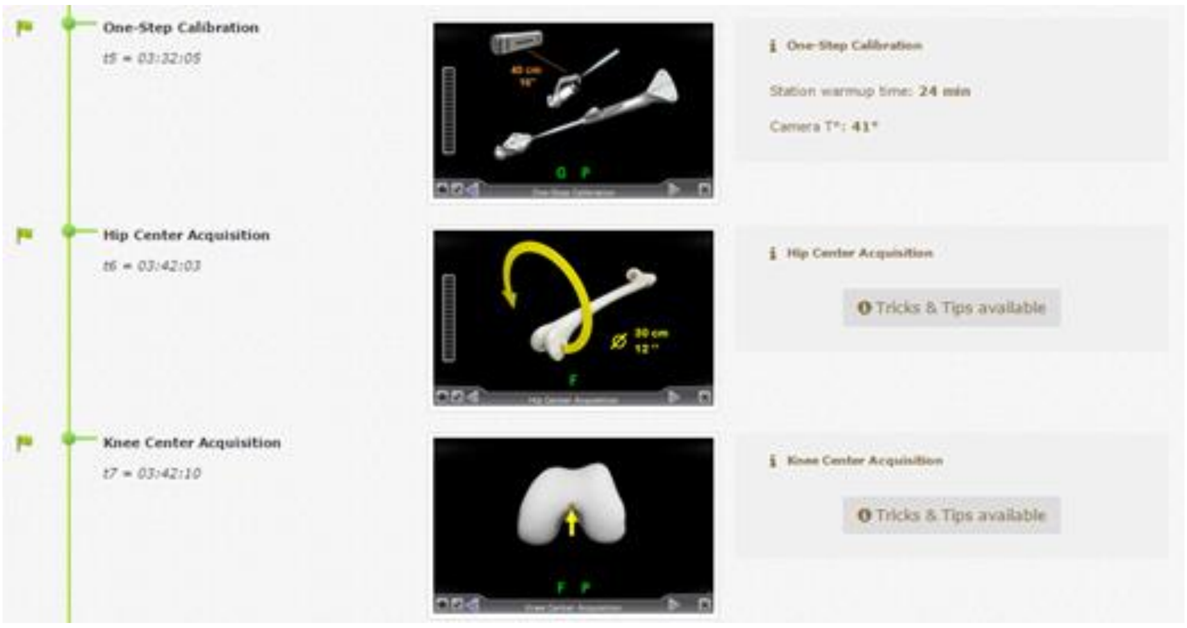
Once the MCPS is certified for some market and actually used, BO still performs activities which are highly relevant to the MODMED project since they are critical to ensure patient safety and could be supported by our tools. BO presented a few studies that were done thanks to the collected traces, but also examples of studies that could not be completed by lack of tools or method.

Misuse surveillance

As a general usability principle, BO chooses to only block the user when the action would undoubtedly have dire consequences for the patient. Consequently, it is important to study how the MCPS is used in the market to detect misuses. This allows warning users about potential problems and advising them on how to avoid these problems in next surgeries. On the other hand it may denote usability problems that should be tackled by BO. In any case, BO feels this is a very relevant activity to improve MCPS safety and effectiveness.

A typical example is to verify that TKA is used within intended operating temperature range because it affects the camera accuracy. TKA itself checks this prerequisite environment condition and the user is warned about possible accuracy problems but he is left responsible for using it or waiting for camera warmup. The same requirement is checked on surgery reports that the surgeon can consult on a dedicated website. This is a successful example of using traces to educate users without taking on R&D resources. Unfortunately, this is an exception and most basic misuses are not detected resulting in 1/ user frustration and 2/ involvement of rare and expensive R&D resources to diagnose trivial problems.

BO also studied whether it was possible to detect the use of a leg holder (which is incompatible with TKA use) based on knee and hip center distances. Another study was made on 1000 traces to detect whether the user should redo some acquisition because the pointer was not sticking to the bone. The problem with these studies is that it is difficult to establish a threshold on hip centre gesture amplitude or point clouds metrics, etc. and BO feels like they lack powerful tools to perform more studies and implement more checks on surgery traces.
Example of a real surgery report with a TKA misuse detected during Acquisitions and links to appropriate warnings and advices (Tricks and Tips):


One-Step Calibration
t5 = 03:32:05

One-Step Calibration
Station warmup time: 24 min
Camera T°: 41°

Hip Center Acquisition
t6 = 03:42:03

Hip Center Acquisition
Tricks & Tips available

Knee Center Acquisition
t7 = 03:42:10

Knee Center Acquisition
Tricks & Tips available

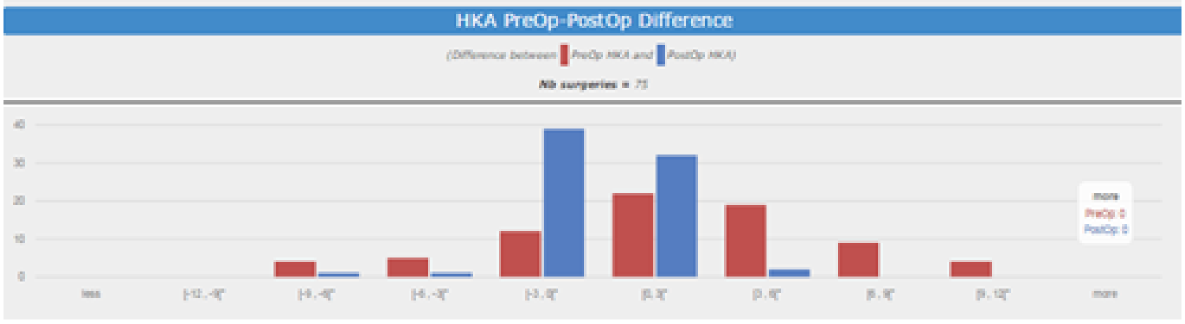
Device performance surveillance

It is more difficult to test camera accuracy in the market since the MCPS does not have external information to check its accuracy. It may be possible though to observe accuracy problems by observing the reported geometry of each tracker.

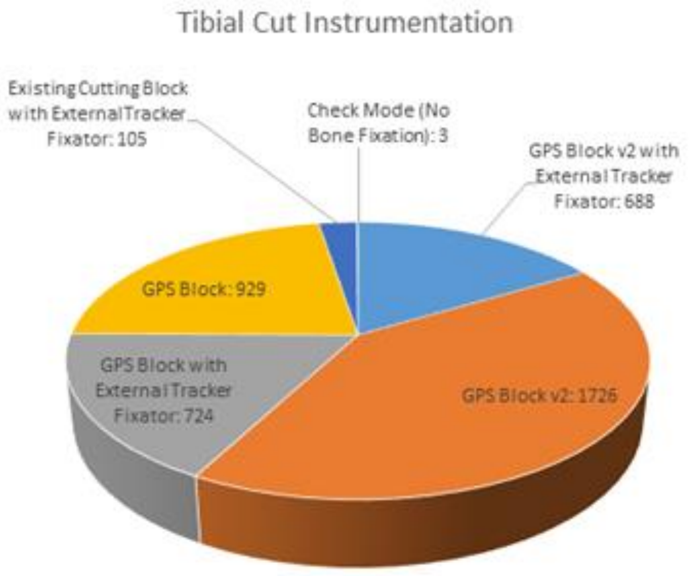
Usage studies

Finally, an important outcome of surgery reports is to give users and manufacturers a feedback on TKA usage.

For instance, each surgeon can see some anatomical metrics about his surgeries (Hip-Knee-Ankle angle before surgery in red, after surgery in blue):



On the other hand, the manufacturer can see which components are actually used to optimize the set of instruments manufactured:



R&D teams get information on the deployment of new software and hardware versions and they can study how this affects surgeries time. For instance, the GPS Block **version 2** (with 3 orientation screws and 1 long translation screw) did improve the time to position cutting guides.

Actors

From the analysis of the aforementioned activities, it appears that the following actors may use MODMED tools and methodology directly or indirectly:

SW Engineers

They are the only ones who could author properties since it requires in-depth knowledge of the MCPS, ability to insert trace appropriate points and sufficient technical background to learn how to write formal properties of traces.

They could leverage the power of properties which are more general than existing unit tests by verifying them in more occasions: verification, production, real surgeries.

They will need a better trace library to produce Structured Traces from existing trace points and be able to easily insert new trace points to trace internal values and control flow without incurring performance hits.

Quality Engineers

Checking TKA requirements during validation and real surgeries would help them fulfill medical device manufacturers' obligations and make them promoters of MODMED tools if and only if properties are kept readable.

Testers and Production Engineers

The verification of manual test results could be automated by properties of traces provided testers can log information on test cases performed and this information can be merged with TKA traces.

Exactech Representatives

Exactech representatives assist surgeons in using TKA. Before the first use, they adapt TKA workflow to the surgeon's one using a SurgeonProfiler application that will save the surgeon's "profile" in a USB stick that they will plug onto the ExactechGPS station. During interventions, they may check that the surgeon understands how to use the product and answer questions on how to use TKA.

Periodically or on-demand, they are asked to collect TKA traces on a USB stick to send them on a dedicated GPSWeb website. BO performs a few checks on the traces which may result in advises that the surgeon will see when he connects to GPSWeb.

BO would like to automate more trace analysis to better help users. Representatives could use these tools to better answer the surgeon questions when an operation did not perform as planned.

Other MCPS industry practices

Based on MMM experience with other MCPS manufacturers, it looks like BO activities are very similar to other manufacturers activities although they may use different terms. For instance, some MMM partners aligned their Quality Management System terminology to the one used by FDA in their guidances. In FDA terminology, TECSP are named "Software Requirements Specification" (SRS) and contain similarly technical requirements, covering all user level requirements thanks to requirements traceability.

4. Requirements studied

The design documents were thoroughly analysed to identify the product requirements that could be verified, and some representative traces were studied to check whether the corresponding trace properties could actually be verified.

Moreover, in order to target MODMED tools to goals that are deemed important by users in the industry and facilitate their adoption, BO was asked to express whether the discussed properties were interesting from the industrial standpoint.

Inappropriate sources of requirements

User level requirements

The requirements listed in the MARSP and FUNSP are so general that they were not deemed interesting for the MODMED project. Also, the Validation activity performed by end users seemed too focused on the intended use to present challenging cases not already tested by the Verification activity. We will see below that the Post-Market Surveillance is an activity triggering user level requirements more interesting for the MODMED project.

Workflow requirements

BO invested a lot of effort to implement the desired surgery workflow as a hierarchical state machine driving the whole UI. This state machine has been studied in detail by MODMED partners because it looked like a good source for properties of traces and its size was unusually large for controlling a UI. Some properties were formalized and verified using prototypes. But the result was usually showing problems with the formalization rather than problems with the SW. For instance, a property stating that “user should pass at least 5 seconds in each state” showed that the property should be further detailed to account for technical states that automatically trigger an exit transition and falsified the property.

Moreover, the state machine is dynamically changed to take into account user interactions like “camera reconnection” or “redo acquisitions” that are not in the normal workflow and these changes are not totally traced in the current version (in spite of BO goal to trace all state machine changes). Finally, the transitions’ conditions are only visible in the source code (making the state machine appear as non-deterministic) and duplicating them in properties would be a lot of work.

BO conclusion is that using the state machine to derive formal properties of traces would be a duplication of the effort already spent on design with a low probability of detecting real problems. MODMED partners are very sensitive to this appreciation since a major challenge of the project is the adoption of its tools by the industry.

On the other hand, it may be useful to verify:

1. general properties of the state machine implementation or specification (infinite loops or dead-ends like the one found during the hands-on session), or in the context of

multi-threading (if some developer directly accesses the state machine from another thread bypassing the expected inter-thread communication channel)
2. that the state machine instantiated by TKA actually respects the surgeon's workflow preferences defined in the SurgeonProfiler tool and general rules like : “dynamically added workflows start and end in the same state of the normal workflow”

GUI requirements

Ensuring the GUI displays accurate and timely information to the surgeon is an important requirement. Unfortunately, current traces are somewhat limited in this area. User actions are traced but reactions of the GUI like audio feedback are not. A few screenshots are taken at each workflow step but the timely updates of GUI components are not (imagine the GUI freezes while the surgeon is adjusting cutting guide screws and cut bones with the illusion that the guide is correctly positioned).

It would be important to be able to extract information from screenshots to automate the verification of some explicit or implicit requirements:

- text displayed (some languages may result in important information being truncated)
- colours used (alerts may use well-defined colours in other applications than TKA)
- anatomical orientation (a left knee may show as a right knee if using an incorrect space transformation)
- etc.

However, this requires skills that are outside MODMED partners' specialties. So, it was decided to first tackle this problem by providing facilities in WP2/D2 trace library to trace GUI behaviour without the risk of delaying GUI updates.

Since this approach will not detect problems in system GUI components like OpenGL, LIG will study ways to extract information from screenshots.

Accuracy requirements

During our discussions, we realized that some requirements mentioned in the Scientific Document like “The precision of the computed hip centre is less than 1 mm” would not be verifiable stricto-sensu using MODMED tools which are based on MCPS traces because the absolute precision requirement requires ad-hoc test benches. What MODMED tools can verify using only information coming from the MCPS is a weaker version of this requirement: “All computed hip centres are located in a 1 mm sphere” which tells something on TKA (inconsistent measurements probably indicate a misuse or failure of TKA) but nothing on the ground truth.

Real time requirements

One surprising conclusion of our analysis is that requirements on the “real” (continuous) time are an exception in TKA and they are usually “soft” real time requirements, in the sense that the time constraints may occasionally be violated without harm to the patient. Effectively, very few hard real time requirements were found in TECSP like F4-T4 “The system detects the new tracker in less than 10 seconds” while other real time requirements remain elusive like F17-T4 "Each click leads to an immediate change of GUI" or F27-T1 "The position of the pointer is displayed in real time over the scheme of the bone".

The main reason for being elusive is the necessity to distinguish such "Usability" requirements from critical performance requirements. For instance, the FDA may unduly interpret F4-T4 as a critical performance requirement (whereas it is not) and investigate: how it was measured, how the design helps fulfilling it, etc.

Another reason is the difficulty to establish a threshold falsifying the property because of our limited knowledge of the real system performance and environment. It is not clear how MODMED could help establish such thresholds so as to obtain useful properties giving an easy-to-understand "Passed or Failed" result. What is clear is that MODMED tools must not only inform on property failures but also on the reasons for failure.

> NB: The Scientific Document mentions other "soft" requirements like "Each point registration step should not occur more than 2 times in average". This kind of properties will be named Usage properties (including Performance properties) in the MODMED project.

Unit tests

Unit tests are a way to implement some aspects of a more general requirement, so we wondered whether they would represent a relevant source of requirements. However, BO feels like SW design patterns (State Machines, "Referenced" geometry classes, etc.) and Unit tests are adequate to independently test SW components and do not feel the need for using new tools at the implementation level (except tests coverage tools). Effectively, BO now experience much more problems with integrated HW components than homemade SW components. So, apart from a few examples, we did not further study unit test programs, neither did we study DETSP documents.

However, it will be interesting to see if some unit tests can be rewritten as the combination of a unit testbed and a property of traces because 1) they may be easier to write with a higher-level MODMED DSL and trace library and 2) they would be more general and could be verified in conditions that the tester did not anticipate as problematic, for instance during the "Verification" activity, or even more interestingly, combined with test assessment and generation tools.

As an example, the VVLabel::test_01 above is specific to a single input which 1) may not represent all problematic cases of the initial implementation, and, despite all Quality Assurance activities, 2) may become incomplete after the implementation changes. Another example is the integer input chosen in geom::test_Constructor_02 which is not representative of floating point values actually used in TKA.

This approach will require complementing TKA traces, though. Effectively, current traces focus on capturing what happened in the environment and the value of intermediate variables usually checked by Unit tests is not traced. Replaying traces was examined as a way to use existing traces and BO experimented it with a prototype but replaying up to the application state producing the desired intermediate values seemed impractical. Moreover, the trace library should provide facilities to define scopes in the trace that would be used to restrict the property to specific unit tests.

Relevant and representative requirements

In the course of MODMED meetings, BO realized it would be very interesting to have better tools to support the Verification activity because 1) it is still very labour intensive, 2) it includes hardware components that are harder to test effectively and 3) its accuracy relies on the tester experience. Effectively, performing the test cases and interpreting the results frequently requires specific know-how that cannot be fully described in the TECSP as in F1-T2 above: “The VirtualKeyboard allows filling the form as expected”.

As a result, a list of 43 requirements in 11 “functions” described in TECSP was determined as relevant based on the interest expressed by the manufacturer and the MODMED project challenges and is given in Appendix 1.

From this list, a short list of 15 properties were deemed “representative”, based on the Dwyer pattern used, the type of event data, and the operations performed on event data:

P1. The trace contains a step “redo acquisitions”.
The “redo acquisition” step allows the surgeon to correct his previous acquisition. It is not part of the standard procedure flow and, therefore, interesting to detect.

P2. The temperature of the camera stays within a given interval.
If used in proper conditions, the camera temperature should not deviate from the range where its precision is guaranteed.

P3. The distance between pairs of hip centres is less than d.
This property asserts that the algorithm computing the hip centre is stable and TKA was used correctly (no leg-holder for instance).

P4. The distance between the hip centre and the knee centre is greater than d.
A violation of this property could reveal an abnormal positioning of the patient or the sensors.

P5. If the medial malleolus is farther from the camera than the lateral one, a warning is issued.
A violation of this property could reveal that the 3D camera was installed on the wrong side of the patient.

P6. The user never skips a screen.
The surgeon is expected to spend sufficient time to appreciate the information showed at each step of the procedure but in our experiments, it was necessary to distinguish this assumed property from the required property that the “next” button does not interpret a single user touch as multiple touches on subsequent screens.

P7. The acquisition of a point succeeds if and only if the probe is stable.
If the surgeon moves the probe tip during an acquisition, it should not be accepted.

P8. The protocol “redo acquisitions” proposes only already performed acquisitions.
The system should not offer the user to redo acquisitions that were never performed.

P9. Detecting a new tracker produces a dialog asking for replacement confirmation.

P10. The state TrackersConnection is unreachable until the camera is connected.
The system should not reach a state dependant on the camera until the camera is connected.

P11. A replaced tracker is not used until it is registered again.
If a tracker is replaced, the system should not try to use it until it is registered again.

P12. The action "previous" cancels the current point cloud acquisition.
Acquiring a points cloud takes a few seconds and can be cancelled. In this case, the current acquisition should not succeed.

P13. All the necessary trackers are seen before entering the state TrackersVisibCheck.
To proceed, the system requires a set of trackers depending on the profile in use. All these trackers should be seen at least once before entering the state TrackersVisibCheck. Note that if we go back to the beginning and change the profile, the trackers already seen do not have to be seen again.

P14. On the trackers connection screen, a tracker is shown if and only if it is necessary.
Only the set of required trackers is shown to the user.

P15. In the state TrackersConnection, not detecting any new tracker for 2 minutes produces an error message.

These properties will be used in further documents to guide the design of the DSL and later validate MODMED tools and methodology. A detailed formalisation of those will be given in WP1 deliverables as examples.

5. Analysis of representative properties

The properties listed in the previous section belong to three categories: Required, Assumed and Usage properties.

Required properties

These properties must be ensured by the TKA system, and more precisely by the TKA software. These properties correspond to requirements on the software. For example, this is the case of property P7 *“The acquisition of a point succeeds if and only if the probe is stable”* which requires the TKA software to check stability of the probe before validating the acquisition of a point. 11 out of the 15 properties correspond to this kind of properties.

Checking these properties on the traces should always succeed, otherwise it would reveal a failure of the system. Ideally, these properties should be proven on the system. This is why we refer to these as *“required” properties*. In the MCPS industry, a more practical approach would be to provide verified traces as evidences that they are fulfilled by the product, and not full proofs.

A special case is when a property is specific to a particular test case. For example in test case F17-T2, the expected values for the computed Cut Height, Varus and Slope could be expressed in a trace property, that will only be checked on traces corresponding to this test case. Such trace properties will thus express the oracle for a given test case.

Assumed properties

These properties should be ensured by the environment of the TKA system. They appear as assumptions on the behaviour of this environment. If the environment fails to fulfill these properties, the behaviour of the TKA system may be affected. For example, this is the case of property P2 *“The temperature of the camera stays within [l,u]”.* If the temperature is outside this range, the precision of the camera may be affected.

If the environment does not behave as expected, the TKA system is designed to stop assisting the surgery in the worst case. In any case, it ensures that the information displayed to the surgical team is correct, or stops displaying information if its correctness is not guaranteed.

Checking these properties on the traces is expected to succeed because the surgeon and his team are expected to use the system in the prescribed conditions. If one of these properties is not satisfied, it may be an explanation for difficulties arising during the surgery and it does not necessarily reveal a defect of the system.

These assumptions on the behaviour of the environment will be referred to as *“Assumed” properties*.

Usage properties

Properties can also be used to identify the set of traces which satisfy these properties. They can be checked to understand the way the system is used.

For example, this is the case of property P1 *“The trace contains a step ‘redo acquisitions’ “*. The *‘redo acquisition’* step was triggered by the surgeon and may reveal that it is difficult to have all acquisitions right at the first attempt, or that the surgeon is not trained enough to use the system. Checking these properties helps understand the way the TKA system is used, but does not reveal a particular failure of the system or its environment. It can be exploited to identify potential evolutions of the TKA system (e.g. efforts should be done to facilitate acquisitions). This is why we refer to such properties as *“Usage” properties*.

Statistics can be computed on the number of traces satisfying a given usage property. At longer term, the DSL might also be used to express *“quantitative usage properties”* reporting quantities instead of Boolean values like a number of occurrences of an event or the duration of a step or the distance between anatomic points, etc. The result of the property on a corpus of traces could then generate a distribution of values rather than a ratio and this would help establish thresholds on, say, acceptable camera temperatures, hip centre gesture amplitude or speed, etc..

When should these properties be checked?

Traces can be produced in four contexts: development, qualification, production and exploitation.

The development context corresponds to all activities which will create or modify the software or the system. They correspond to the initial development, but also to corrections during the maintenance phase and evolutions of the system. In the development context, traces will be produced during tests. They will be used to evaluate the correctness of the system and hence required properties will be the most important ones at this stage.

The qualification context follows development activities, it is aimed to demonstrate the correctness of the system and validate assumptions on its environment. Here again the focus will be on required properties, but in the qualification context, these properties are expected to be fulfilled by the system under qualification. The qualification phase also involves “Acceptance tests” typically performed by surgeons on cadavers using a successfully verified MCPS. During these acceptance tests, one can expect that required properties will be satisfied but the focus will be on assumed and usage properties.

Since the TKA system is composed of software and hardware, it involves a production phase where the system is manufactured and tested. Here the tests are aimed at checking the hardware for manufacturing defects. The properties checked at this stage are required properties because they aim to check the system, and not its environment. Still, while in the development and qualification contexts failure could result from software defects, in the production phase, the software should be correct and failures only reveal hardware defects, or incorrect execution of the tests by the tester.

The exploitation context corresponds to the operation of a qualified system during a real surgery. In this context, required properties should not fail. Checking these properties on traces of numerous surgeries brings additional evidence of the quality of the qualified system.

Assumed and usage properties are mostly relevant to the production context. Assumed properties correspond to assumptions on the way the system is operated. Checking the assumed properties on the traces of real surgeries will help detect cases where the environment of the system was not adequate. Detecting such traces may bring explanations on why something did not proceed smoothly, or require for more robustness of the system against the failure of these properties.

Usage properties are aimed at understanding how the system is used in normal operation. It is thus aimed at analysing traces exhibiting the real behaviour of medical teams.

Properties classification

Each property can be characterized by the number of different events it involves and whether it:

- is *parametric*, *i.e.* it involves event parameters,
- is *temporal*, *i.e.* it constrains the order of two or more occurrences of events,
- applies to a restricted scope of the trace,
- has geometric predicates on data,
- has GUI predicates on screenshots,
- involves physical-time, and
- is a required, assumed or usage property.

The following table synthesizes the classification of the 15 properties presented in the previous section:

Property	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
No. of event types	1	1	1	2	3	2	2	2	2	2	3	3	4	2	4
Parametric		✓	✓	✓	✓	✓	✓	✓	✓		✓		✓	✓	✓
Temporal						✓	✓	✓	✓	✓	✓	✓	✓		✓
Restricted scope								✓	✓	✓	✓		✓		✓
Geometric predicate			✓	✓	✓	✓									
GUI predicate														✓	
Physical time						✓									✓
Required			✓		✓		✓	✓	✓	✓	✓	✓	✓	✓	✓
Assumed		✓	✓	✓		✓									
Usage	✓					✓									

In the following, we will detail this classification for a couple of properties.

Let us consider property P3, stating that the distance between pairs of hip centres is inferior to a given threshold. It involves several events of a single type, reflecting the acquisition of a new hip centre and that is parametrized with the acquired point. These events parameters must satisfy a geometric constraint. A violation of this property could indicate that the patient was not installed as expected or, if the patient was correctly installed, that the algorithm computing the hip centre is not stable. Thus, property P3 is both a required and an assumed property. It would be important to report whether the assumption or the requirement was violated by a particular trace in order to correctly interpret it but it may not always be possible. For instance, we would like to complement P3 with something like “assuming the patient was installed on the correct side”, but this information is not in the traces.

Property P12 is a required property stating that the action “previous” cancels the ongoing point cloud acquisition. In other words, triggering the action “previous” during an acquisition prevents this acquisition from succeeding. This property involves three different event types, reflecting the action “previous”, the beginning of an acquisition, and the success of an acquisition. No event parameter is needed, which makes the property nonparametric. However, it is temporal since the occurrence of the “acquisition success” event is constrained by other event occurrences.

Most of the results of this classification are in accordance with the expectations of the MODMED project: the properties are very diverse and rely heavily on data parameters. On the contrary, the “real” time is rarely involved in the identified properties despite our expectations.

6. Partial conclusions for further MODMED research

The study of the TKA MCPS, design documents and surgery traces lead MODMED partners to focus on requirements which were the most relevant from the industrial standpoint. These requirements mainly come from BO TEChnical SPecifications (TECSP) document and are expressed in informal English as a table of test conditions and expected results. Other sources of requirements that MODMED tools may support are MCPS usage studies and, to a lesser extent, selected unit tests.

These requirements were rewritten as properties of traces and classified. The most fundamental result of this classification was to distinguish between Required properties, Assumed properties, and Usage properties since the falsification of such properties by a trace has very different meaning and consequences. It may help to make this distinction in the WP1 DSL. This may take the form of informal text associated to the properties.

Usage properties are quite different from the two others but they are deemed important by BO. The DSL should be kept open to the addition of quantitative usage properties to extend its applicability to industry problems.

Other aspects of the classification lead us to select 15 representative properties to further guide the MODMED research and tools development. Although supporting physical time was initially planned as an important research direction of the project, the requirements study shows that it is not as important and the MODMED project will reprioritize it consequently.

Requirements analysis showed that formalising MCPS requirements as properties of traces will remain a delicate task requiring a good understanding of seemingly similar things like Required and Assumed properties along with technical skills. Thus, it is important and challenging to facilitate authoring such properties by SW engineers. Keeping WP1 DSL properties readable by less technical people like Quality engineers will be important but should not complicate writing them. Moreover, the DSL should be designed in conjunction with WP5 tools to help understand them in the light of existing traces or using simple synthesized traces satisfying or falsifying a property.

WP3 tools should provide elements to help SW and Quality Engineers understand why a particular trace does not satisfy a given property.

Finally, we believe that writing unit tests using properties of traces is interesting but it is not feasible because existing traces lack intermediate values and control flow. Thus, WP2 will study why this data was not traced and provide appropriate methodology and tools to significantly increase the level of detail of traces, especially in the qualification context where it is possible to manage performance and size hits.

Appendices

1. Requirements derived from “TEChnical SPecification”

F4 - Connect Camera and Tracker

- 4 trackers of different types are seen before leaving the state *TrackingConnection*
- The procedure should not start before connecting the camera
- If the camera is disconnected, it must be connected again to continue the procedure
- In the state *TrackersConnection*, not detecting any new tracker for 2 minutes produces an error message
- Detecting a new tracker produces a dialog asking for replacement confirmation
- A tracker that was refused must not be used
- A tracker that was replaced must not be used
- Disconnecting the camera produces an error message
- Accepting to reconnect the trackers starts a special procedure
- When the trackers reconnection procedure is successfully completed, the error message disappears
- Triggering the “previous” action during the connection procedure causes the system to go back to the system into the profile loading screen
- Refusing to connect the trackers when prompted causes the system to go back to the system into the profile loading screen
- On the connection screen, the tracker "F" is shown if and only if the profile includes functions on the femur
- On the connection screen, the tracker "T" is shown if and only if the profile includes functions on the tibia

F5 - Calibrate the pointer tip

- The distance between the probe tip and the calibration cone of a tracker must be inferior to a given threshold
- If the calibration fails, an error message is displayed

F7 - Acquire single point with the pointer

- Acquiring a point succeeds if the probe is stable during the acquisition
- Acquiring a point fails if the probe is moved
- After a successful point acquisition, the system automatically moves to the next step

F8 - Acquire cloud of points with the pointer

- After a successful points cloud acquisition, the system automatically moves to the next step
- Triggering the “previous” action cancels the ongoing acquisition

F9 - Compute Ankle Centre reference

- If the lateral and medial malleolus are reversed, an error is issued and the acquisition is refused
- After a successful acquisition, the system automatically moves to the next step

F12 - Compute Hip Centre

- After a successful acquisition, the system automatically moves to the next step
- If the movement amplitude is too low, an error message is displayed and the acquisition is refused
- The acquisition succeeds if all the results are inside a sphere of 7mm diameter
- If the computation of the hip centre fails, an error message is displayed and the acquisition is refused

F15 - Acquire ligament balancing in flexion

- With the test profile “p1”, the following values are displayed: Flexion value, Rotation value (Not Varus/Valgus), Anatomical Gaps
- With the test profile “p1” and the knee is in flexion and internal rotation, the displayed label for the rotation is "Int" and the value is strictly positive
- With the test profile “p1” and the knee is in flexion and external rotation, the displayed label for the rotation is "Ext" and the value is strictly positive
- With the test profile “p2”, two rotations values are displayed
- With the test profile “p4”, no rotation values are displayed
- If the flexion of the tibia is outside of the range [80-100]°, an error message is displayed
- The acquisition is refused is the tibia is moved during the digitization

F16 - Plan the tibial cut

- With the test profiles “p[1,2,3]”, the following information are displayed and can be adjusted: slope, varus/valgus, cut height for medial and lateral side
- Angular values and cut heights are updated following the clicked arrow direction

F17 - Navigate the tibial cut

- With the test profile “p1”, the following information are displayed: slope, varus/valgus, cut height for medial and lateral side

F18 - Digitize the tibial cut

- If the cutting guide is moved for more than 1mm on cut height and 1° on slope or flexion, the acquisition is refused
- After a successful acquisition, the system automatically moves to the next step

F22 - Plan the 4in1 femoral cut

- With the test profile “p1”, the following information are displayed and can be adjusted: notching, rotation, size. Posterior cut heights (medial & lateral) and rotation are also displayed.
- Angular values and cut heights are updated following the clicked arrow direction
- The default cut proposed is conform to the profile
- A ruler is displayed