S. Boll, S. Maaß & R. Malaka (Hrsg.): Workshopband Mensch & Computer 2013
München: Oldenbourg Verlag, 2013, S. 201–206

TUI-Framework zur Integration be-greifbarer Objekte in interaktive Systeme

Oliver Belaifa, Johann Habakuk Israel

Virtuelle Produktentstehung, Fraunhofer IPK Berlin

Zusammenfassung

Dieser Beitrag stellt ein technisches Framework vor, das es Entwicklern erleichtert, verteilt und über Betriebssystem-Grenzen hinweg auf Ein- und Ausgabegeräte zuzugreifen. Be-greifbare Objekte, die hardwareseitig verschiedene Sensor- und Aktuator-Technologien nutzen, können mit Hilfe des Frameworks in einfach zu handhabende Software-Objekte gekapselt werden. Ursprünglich wurde das Framework entwickelt, um die Integration von be-greifbaren Objekten und Werkzeugen in virtuellen Umgebungen zu unterstützen. Das Framework findet derzeit aber auch Einsatz außerhalb immersiver Umgebungen. Dieser Beitrag skizziert den aktuellen Stand des Frameworks, das bereits zuvor vorgestellt wurde, und zeigt neue Anwendungsbeispiele und neue Geräte-Schnittstellen.

1 Einleitung

Das vorgestellte TUI-Framework wurde mit dem Ziel entwickelt, die Entwicklung von be-greifbaren Objekten und Werkzeugen in virtuellen Umgebungen zu erleichtern. Diese benötigen meistens ein Trackingverfahren, um im Raum erfasst werden zu können, und Sensoren, z. B. Knöpfe und Schalter, mit denen der Benutzer Aktionen auslösen kann. Außerdem ist es insbesondere für be-greifbare Schnittstellen wichtig, auch über einen Rückkanal von der Applikation zum Werkzeug zu verfügen, beispielsweise für Kraft-rückkoppelnde Geräte. Diese und weitere Funktionen werden durch das TUI-Framework realisiert (Israel, Belaifa, Gispen & Stark, 2011). So können mithilfe des Frameworks Softwareobjekte (TUIObjects) entwickelt werden, die Informationen aus Eingabe-Kanälen unterschiedlicher Interaktionsgeräten kapseln und zusätzlich in der Lage sind, Informationen an Ausgabe-Geräte zu senden. Die Entwicklung des TUI-Framework wurde 2009 im Rahmen des BMBF-geförderten Projekt AVILUSplus begonnen (AVILUSplus, 2011) und wird jetzt im Rahmen des EU-geförderten Projekts VISIONAIR weiterentwickelt (Visionair, 2013). Das Framework ist als Open-Source-Projekt samt Beispielimplementierungen verfügbar unter https://github.com/fraunhoferipk/tuiframework. In diesem Beitrag soll die veränderte Archi-

tektur des Frameworks, die sich daraus ergebenen neuen Möglichkeiten und neue integrierte Interaktionshardware vorgestellt werden.

2 Der TUI-Server

Der TUI-Server dient als zentrale Datenaustauschstelle im Framework. Er wird mit einer Konfigurationsdatei verwaltet. Diese Datei enthält Einstellungen und Definitionen der verwendeten TUI-Entitäten sowie den zugehörigen Datenflussgraphen. Die TUI-Entitäten gliedern sich in drei Klassen. *TUI-Devices* sind Abstraktionen für die Hardware, sie repräsentieren z. B. Trackingsysteme oder Force-Feedback-Geräte. *TUI-MSPs* (Multi Stream Processors) manipulieren oder synchronisieren Datenströme. *TUI-Objekte* abstrahieren konkrete Interaktionsobjekte, also beispielsweise Werkezuge, die optisch getrackt werden und über physische Schalter verfügen. Die Konfigurationsdatei liegt im XML-Format vor und wird vom TUI-Server beim Einlesen mit Hilfe einer XML-Schema-Definition validiert.

Das manuelle Editieren der Konfigurationsdatei ist eine häufige Ursache von Fehlerquellen bei der Inbetriebnahme eines TUI-Servers. Um dem vorzubeugen, wurde ein Konfigurationstool programmiert, welches dem Nutzer eine schnelle graphische Konfigurierung des TUI-Servers ermöglicht (Abschnitt 7).

3 Plug-in-Architektur

Der TUI-Server verwendet in der Regel mehrere Module, um ein TUI-Szenario zu realisieren. Diese können in einem TUI-Server eingebettet sein oder dem TUI-Server als Bestandteile gemeinsam genutzter Code-Bibliotheken zur Verfügung stehen. Letzteres erfordert die Unterstützung einer sogenannten Plug-In-Architektur. Sie erlaubt die Kompilierung eines generischen TUI-Servers, welcher die zusätzliche Funktionalität der weiteren Module über die TUI-Plug-Ins erlangt. Für die Unterstützung der Plug-In-Architektur bietet das TUI-Framework vorgefertigte systemspezifische C++-Klassen für Windows, Linux und MacOS. Beim Start des TUI-Servers mit Plug-In-Architektur wird in einem Verzeichnis nach gemeinsam genutzten Code-Bibliotheken, nach TUI-Devices und TUI-MSPs sowie gegebenenfalls neuen Datentypen gesucht. Diese werden automatisch registriert und bereitgestellt und stehen daraufhin im TUI-Framework zur Verfügung.

4 Flexibler Einsatz des TUI-Frameworks

Für ein TUI-System wird oftmals Hardware verschiedener Hersteller für die Realisierung der TUI-Interaktionsobjekte benötigt. Beispielsweise kann ein Stift für das Skizzieren im Raum einen Marker für das optische Tracking der 3D Position und Orientierung, einen Drucksensor, eine LED sowie einen Bluetooth-Chip für die Sensorübertragung enthalten (Israel, Wiese, Mateescu & Stark, 2009).

Für die Integration der Hardware können Hürden in Erscheinung treten. Beispielsweise können benötigte Hardware-Treiber nur auf unterschiedlichen Betriebssystemen zur Verfügung stehen oder es können Inkompatibilitäten einzelner APIs untereinander auftreten. Beispielsweise gibt es Hardware-APIs, die Exklusivität innerhalb eines Prozessraums verlangen. Probleme können auch entstehen, wenn die benötigten APIs nur in unterschiedlichen Programmiersprachen zur Verfügung stehen.

Das TUI-Framework bietet für diese Probleme Lösungsmöglichkeiten an und gibt Entwicklern eine große Flexibilität in der Entwicklung interaktiver Szenarien (vgl. Abbildung 7):

- Das TUI-Framework wurde plattformunabhängig programmiert und kann unter Windows, Linux und MacOS verwendet werden. Somit kann bei Betriebssystem-Einschränkungen der Hardware-API ein TUI-Device-Modul unter dem jeweiligen Betriebssystem implementiert werden.

- Das TUI-Framework beinhaltet die Funktionalität der Plug-In-Architektur. Hilfreich ist diese u. a., wenn Hardware-APIs untereinander nicht kompatibel sind oder einzelne Hardware-APIs Exklusivität des Prozessraums erwarten.

- Eine weitere Flexibilitätssteigerung stellt die Möglichkeit dar, mehrere TUI-Server in einem TUI-System zu verwenden. Das erlaubt unter anderem die gleichzeitige Ansteuerung von Hardware auf unterschiedlichen Betriebssystemen. Eine weitere Technik um die Betriebssystemgrenzen zu überwinden ist die Ansteuerung der Hardware-API vom separaten TUI-Device-Modul über eine P2P Netzwerkverbindung. Vorgefertigte Klassen für die Nutzung dieser Funktionalität werden von dem TUI-Framework zur Verfügung gestellt.

- Eine TUI-Anwendung kann aus mehreren verteilten Systemen bestehen. Diese benötigen die Datenströme der TUI-Objekte gegebenenfalls in mehreren Knotenpunkten. Für die Realisierung dieses Anwendungsfalls unterstützt das TUI-Framework die Netzwerkverbindung mehrerer TUI-Clients mit einem oder mehreren TUI-Servern. Kommuniziert wird über das Netzwerkprotokoll UDP im Unicast- oder Multicastmodus.

- Interaktionsgeräte werden durch typisierte Eingangs- und Ausgangs-Ports abstrahiert. Die Typisierung ist für die Applikation hilfreich, damit sie spezifische Handler installieren kann. Der Typ des ankommenden Datenstroms ist bekannt und kann somit einfach verarbeitet werden.

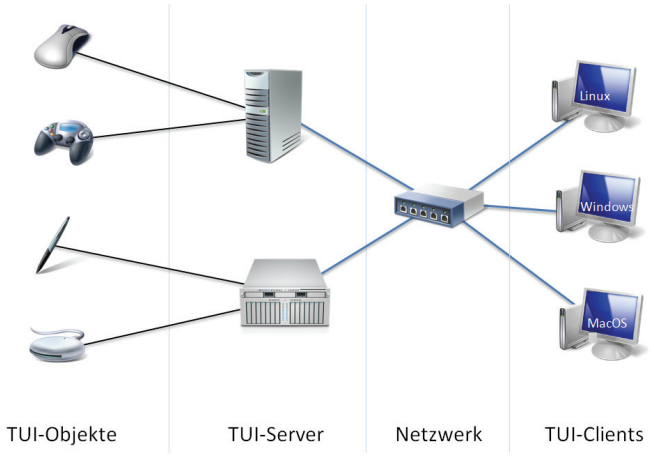

Abbildung 7: Konfigurationsbeispiele des TUI-Frameworks

5 Die Kinect als Interaktionsgerät

Die erste Anwendung des TUI-Frameworks war eine immersive Modellieranwendung in einer Virtual-Reality-Cave. Hierfür wurden spezielle Interaktionsgeräte wie Stifte und Zangen entwickelt. Ihre 3D-Position und -Orientierung werden über ein optisches Trackingverfahren ermittelt und ihre Sensordaten (Knöpfe, Drucksensoren) drahtlos übertragen (Israel et al., 2009).

Um bestimmte Interaktionen der Modellieranwendung berührungslos auslegen zu können, wurde eine Kinect-Tiefenkamera über das TUI-Framework in die Anwendung integriert (Israel & Sniegula, 2012). Hierzu wurde ein Kinect-TUIDevice entwickelt, welches die 3D Positionen der Gelenke (Joints) des Benutzers im Koordinatensystem der Kinect liefert. Um die Positionen in das Koordinatensystem des Trackingsystems zu transformieren, benötigt man die Position und die Orientierung der Kinect im Trackingkoordinatensystem. Hierfür wird ein optischer Marker an die Kinect befestigt. Das Trackingsystem liefert die gewünschte Transformationsmatrix und ermöglicht die Transformation der Joint-Positionen in das Tracking-Koordinatensystem zur Laufzeit.

Für die Interpretation der Joint-Bewegung wurde ein MSP entwickelt, welches mehrere Gesten erkennen kann. Das MSP emittiert bei einer Detektion beispielsweise Zustandsänderungen eines Knopfes. Dies erlaubt es beispielsweise, das optische Head-Tracking durch Informationen aus der Kinect zu ersetzten. Da das TUI-Framework eine Hardware-Abstraktions-Schicht besitzt, muss hierfür applikationsseitig der Programmcode nicht verändert werden, denn die TUIDevices sowohl der Kinect als auch des optischen Trackingverfahrens liefern Ereignisse desselben Typs.

6 Die Multi-Maus-Anbindung

Ein weiteres neues TUIDevice ist ein Multi-Maus-Modul. Es ermöglicht die gleichzeitige Benutzung beliebig vieler Mäuse in einer Applikation. Das Multi-Maus-Modul enthält mehrere Untermodule. Hierzu gehören ein Mouse-Server, ein TUI-MouseDevice und ein Mouse-Demultiplexer. Der Mouse-Server ist nur unter Windows lauffähig und verhindert die Weitergabe aller Mausereignisse an das Betriebssystem, um nicht intendierte Funktionsaufrufe zu verhindern. Erfasst werden alle relativen Positionsänderungen sowie die Betätigung der Mausknöpfe und des Mausrades. Alle Änderungen werden vom Mouse-Server an das TUI-MouseDevice gesendet, welcher sie über die Framework-Ports zu den nächsten verbundenen Entitäten weiterleitet, z. B. zu einem MouseTUI-Objekt.

Ein möglicher Anwendungsfall für die Multi-Maus-Anbindung ist beispielsweise ein Lernprogramm, bei dem sich mehrere Schüler einen PC teilen. Für die Entwickler von interaktiven Systemen ist diese Möglichkeit beispielsweise dann interessant, wenn sie ihre Anwendungen an einem herkömmlichen PC entwickeln, an dem nicht alle benötigten Geräte und Objekte vorhanden sind. Sie können dann jedes Gerät durch eine Maus simulieren, indem sie die Maustasten und –position auf bestimmte Ports des TUIObjects mappen. Später müssen diese Mappings dann mithilfe der TUI-Serversettings lediglich auf die eigentlichen TUIDevices umgesetzt werden, Änderungen im Programmcode werden nicht notwendig.

7 Interaktive haptische Fahrzeugsimulation

Ein neuer Anwendungsfall des TUI-Frameworks ist eine interaktive haptische Fahrzeugsimulation, bestehend aus mehreren Hard- und Softwarekomponenten verschiedener Hersteller (Beckmann, 2013). Der Simulator besteht aus der Motion-Plattform der Firma Festo-Didactic, einem Echtzeitfahrzeugsimulator, einem Multifunktionslenkrad mit Pedalerie und Schalteinheit der Firma Logitec sowie einem Rendering- und Audiosystem (Abbildung 8). Das *Fahrzeug und die externe Umgebung interagierten in einer Co-Simulation.* Für die Intrakommunikation, Steuerung und Synchronisation dieser Komponenten wurde das TUI-Framework eingesetzt. Hierbei musste die Verarbeitung der Daten über das TUI-Framework in Echtzeit geschehen, damit die *Reaktionszeiten des Gesamtsystems gering bleiben, um die Plausibilität der Fahrsimulation zu gewährleisten. Bei der Integration durch das TUI-Framework wurden TUI-Devices der einzelnen Hardware-Module implementiert und TUI-Objekte definiert, welche die einzelnen Module geeignet abstrahieren und der zentralen Steuerungsanwendung zur Verfügung stehen. Einen Ausschnitt der grafisch repräsentierten XML-Konfiguration zeigt* Abbildung 9.

Abbildung 8: haptischer Fahrzeugsimulator

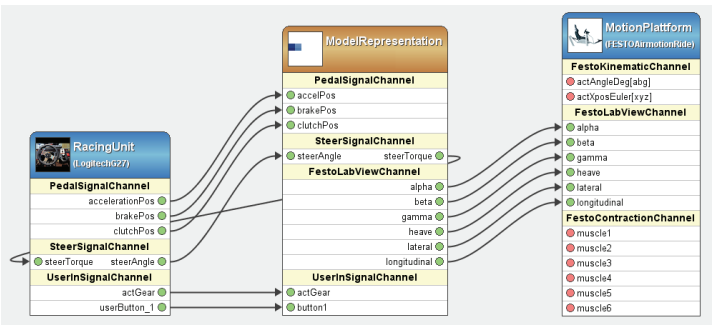

Abbildung 9: Konfigurationsbeispiel des TUI-Frameworks am Beispiel einer haptischen Fahrzeugsimulation (Ausschnitt, aus Beckmann, 2013, S. 59)

8 Schlussteil

Für die Zukunft ist es geplant, das Framework um weitere Geräte zu erweitern, z. B. ein Eye-Tracking-System oder mobile Endgeräte. Weiterhin sollen die Gesamtarchitektur des Frameworks weiter dezentralisiert und die Echtzeitfähigkeit verbessert werden.

Danksagung

Diese Arbeit wurde durch das EU-finanzierte Projekt VISONAIR (262044) für Forschung auf den Gebieten der Visualisierung und Interaktionstechniken unterstützt.

Literatur

AVILUSplus. (2011). *Angewandte Virtuelle Technologien mit Langfristfokus auf den Produkt- und Produktionsmittel-Lebenszyklus*. Zugriff am 8. Juni 2013,von http://www.avilusplus.de

Beckmann, T. (2013). *Interaktive Co-Simulationsmethoden für Virtual Reality Anwendungen „Functional-Drive“ und „Smart-Hybrid-Prototyping“*. Technische Universität Berlin.

Israel, J. H., Belaifa, O., Gispen, A. & Stark, R. (2011). *An Object-centric Interaction Framework for Tangible Interfaces in Virtual Environments*. *Fifth international conference on Tangible, embedded, and embodied interaction ACM TEI'11* (S. 325–332). Fuchal, Portugal: ACM Press.

Israel, J. H. & Sniegula, E. (2012). *Berührungslose und be-greifbare Interaktionen des 3D-Skizzierens*. *Mensch & Computer 2012 – Workshopband: interaktiv informiert – allgegenwärtig und allumfassend!?* (S. 147–153). München: Oldenbourg Verlag.

Israel, J. H., Wiese, E., Mateescu, M. & Stark, R. (2009). *Investigating three-dimensional sketching for early conceptual design—Results from expert discussions and user studies*. *Computers & Graphics*, *33*(4), 462–473.

Visionair. (2013). *VISION Advanced Infrastructure for Research*. Zugriff am 8. Juni 2013, von http://www.infra-visionair.eu/

Kontaktinformationen

Oliver Belaifa, Johann Habakuk Israel
Virtuelle Produktentstehung, Fraunhofer IPK
Pascalstraße 8-9, 10587 Berlin
http://www.ipk.fraunhofer.de/geschaeftsfelder/virtuelle-produktentstehung/