

**System Requirements Specification**

**MultiCastor 2.0**

**V 1.0**

Projekt: MultiCastor 2.0

Auftraggeber: Rentschler & Stuckert

Rotebühlplatz 41/1

70178 Stuttgart

Auftragnehmer: TIT10AID - Team 4 - MCastor2.0

Fabian Fäßler

Filip Haase

Matthis Hauschild

Sebastian Koralewski

Jonas Traub

Christopher Westphal

Rotebühlplatz 41 – Raum 0.10

70178 Stuttgart

Inhaltsverzeichnis

[1. Einleitung 5](#_Toc309157747)

[2. Zielbestimmung 5](#_Toc309157748)

[3. Produkteinsatz 6](#_Toc309157749)

[3.1. Beschreibung des Problembereichs 6](#_Toc309157750)

[3.2. Modell des Problembereiches 8](#_Toc309157751)

[3.3. Beschreibung der Geschäftsprozesse 10](#_Toc309157752)

[3.3.1. Testen der Netzwerkverbindung 10](#_Toc309157753)

[3.3.2. Multicast-Funktionalitätstest 10](#_Toc309157754)

[3.3.3. Belastbarkeitstest 11](#_Toc309157755)

[3.3.4. Fehlersuche 11](#_Toc309157756)

[3.3.5. Testbericht 11](#_Toc309157757)

[3.3.6. Kundendemonstration 11](#_Toc309157758)

[4. Produktfunktionen 12](#_Toc309157759)

[4.1. Use Cases – Übersicht 12](#_Toc309157760)

[4.2. /LUC10/: Send Multicast 13](#_Toc309157761)

[4.3. /LUC20/: Receive Multicast 14](#_Toc309157762)

[4.4. /LUC30/: Analyse Multicast 15](#_Toc309157763)

[4.5. /LUC40/: Configure Settings 15](#_Toc309157764)

[4.5.1. /LUC41/: Save Configuration File 17](#_Toc309157765)

[4.5.2. /LUC42/: Load Configuration File 19](#_Toc309157766)

[4.5.3. /LUC43/: Set/Delete Multicast Group Membership 20](#_Toc309157767)

[4.5.4. /LUC44/: Activate/Deactivate Multicast Group 21](#_Toc309157768)

[4.5.5. /LUC45/: Configure Server Settings 23](#_Toc309157769)

[4.5.6. /LUC46/: Configure Language Settings 25](#_Toc309157770)

[4.6. /LUC50/: Multi-Instance-Ability 26](#_Toc309157771)

[4.7. /LUC60/: Functional Testing 28](#_Toc309157772)

[5. Produktcharakteristiken 31](#_Toc309157773)

[5.1. Systemumgebung 31](#_Toc309157774)

[5.2. Hardwareumgebung 31](#_Toc309157775)

[5.3. Softwareumgebung 31](#_Toc309157776)

[6. Anhang 32](#_Toc309157777)

[6.1. Multiple MAC Registration Protocol 32](#_Toc309157778)

[6.2. Testframework STAF/STAX 33](#_Toc309157779)

[6.2.1. STAF 33](#_Toc309157780)

[6.3. STAX 34](#_Toc309157781)

[Dokumentversionen 35](#_Toc309157782)

# Einleitung

In diesem Dokument (System Requirements Specification / Pflichtenheft) wird näher beschrieben, *wie* die vereinbarten Aufgaben zwischen Auftragnehmer und Auftraggeber zu erledigen sind.

Es beinhaltet zum einen die Erfassung des Problembereiches, sowie der Geschäftsprozesse des Auftraggebers. Zum anderen die Beschreibungen der künftigen Produktfunktionen in Form von Use-Cases. Zusätzlich werden die Produktcharakteristiken wie beispielsweise benötigte Hard- und Software beschrieben.

Zur Sicherstellung der Multibetriebssystemfähigkeit wird der MultiCastor ständig auf den Betriebssystemen Windows, Linux und Mac OS getestet. Um dem Benutzer die unterschiedlichen „Look and Feel“ dieser Betriebssysteme näher zu bringen, haben wir uns entschlossen, bei den Screenshots in unseren Dokumenten verschiedene dieser Systeme abzudecken.

# Zielbestimmung

|  |  |
| --- | --- |
| Vision/Ziel | Beschreibung |
| /V10/ | Der MultiCastor soll um das Protokoll nach IEEE 802.1ak (MMRP) erweitert werden. |
| /V20/ | Der MultiCastor soll mit dem Testautomatisierungsframework STAF/STAX getestet werden können. |
| /V30/ | In Anlehnung an /V40/ des MultiCastors V1.0 soll V2.0 Netzwerkunregelmäßigkeiten nicht mehr nur erkennen, sondern kontrolliert abfangen und gegebenenfalls darauf reagieren können. |
| /Z10/ | Der MultiCastor soll im laufenden Betrieb zwischen Layer-2- und Layer-3-Funktionalität umschaltbar sein |
| /Z20/ | Die graphische Oberfläche soll vereinfacht werden, um neuen Benutzern den Einstieg zu vereinfachen und den Workflow von bereits erfahreneren Benutzern deutlich zu erhöhen. |
| /Z30/ | Die Konfigurationsspeicherung soll überarbeitet werden. Es wird auf ein modulareres System gesetzt, bei dem Programm- und Userkonfiguration getrennt gespeichert sowie geladen werden können. |

# Produkteinsatz

In diesem Kapitel werden die Problembereiche sowie die Gründe der Erweiterung um das MMR-Protokoll dargestellt.

## Beschreibung des Problembereichs

Viele Unternehmen haben riesige Netzwerke, in denen viele Hosts miteinander kommunizieren. Die zumeist eingesetzte Art von Kommunikation ist *Broadcasting*, bei der jeder Host mit jedem Anderen kommunizieren kann.

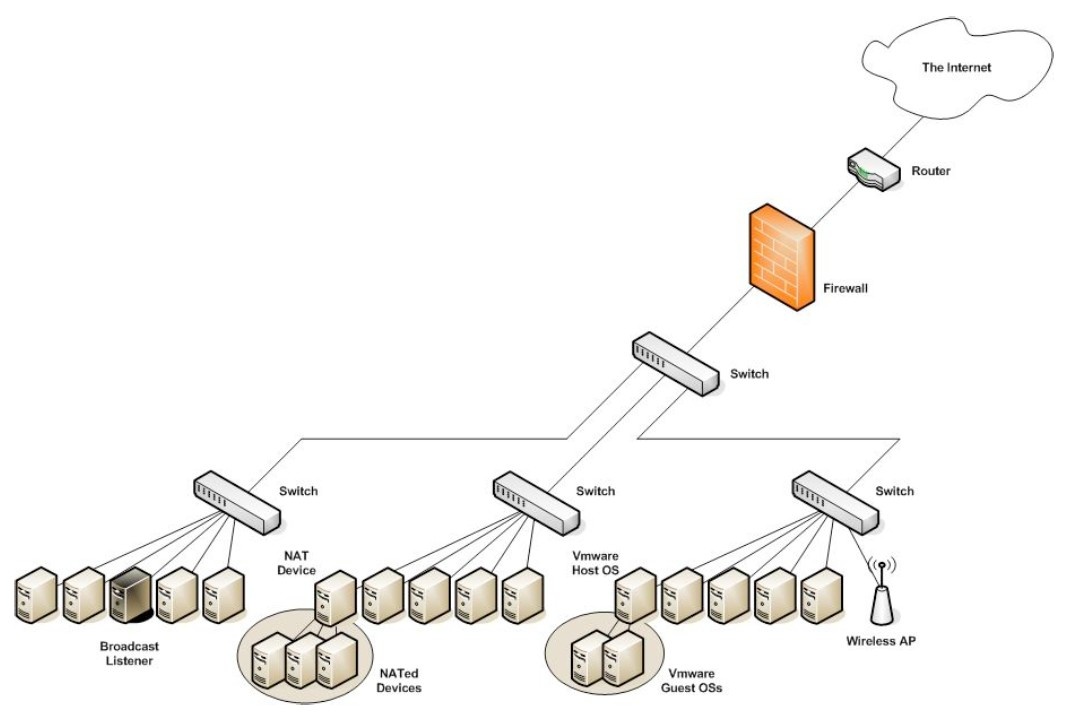


Abbildung : Broadcasting[[1]](#footnote-1),

Dieser Zustand ist nicht immer gewünscht. Sei es, um das Netzwerk zu entlasten oder um Dateien nur bestimmten Hosts zugänglich zu machen.

Abhilfe schafft hier Multicast, denn eben jenes erfüllt die Funktion, eine Nachricht an mehrere User (im Gegensatz zu Broadcast aber **nicht an alle**)im Netzwerk gleichzeitig zu schicken. Der Sender registriert dafür eine sogenannte Multicast-Gruppe bei seinem zuständigen Switch, an dem sich nun Empfänger für diese Gruppe registrieren können.

Der Sender schickt dann seine Daten an den Switch und dieser verteilt die Daten an die registrierten Empfänger.

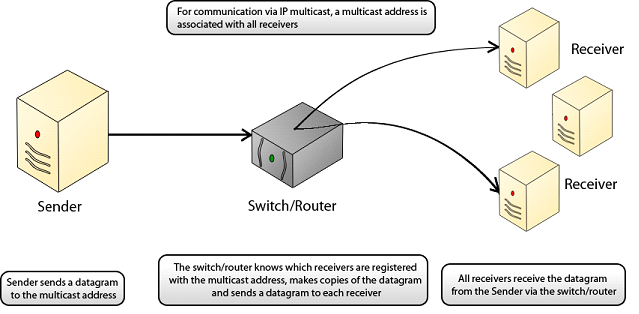


Abbildung : Multicast-Kommunikation im Netzwerk

Der MultiCastor V1.0 nutzt die im Betriebssystem verankerten Protokolle IGMP und MLD, um Multicast auf IP-Ebene (Layer-3) zu testen.

Da ein Switch lediglich auf Layer-2-Ebene (anhand von MAC-Adressen) kommunizieren kann, ergibt sich hierbei der entscheidende Nachteil, dass mehrere Switche ohne Zuhilfenahme eines Routers kein Multicasting betreiben können.

Das Verfahren des Zwischenschaltens eines Routers, um Multicasting betreiben zu können, wird auch als IGMP-Snooping bezeichnet.

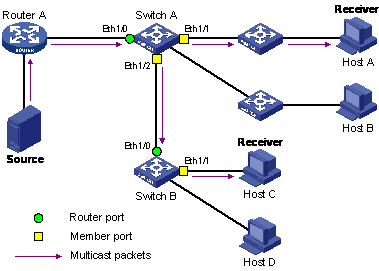


Abbildung : MC in Layer-3[[2]](#footnote-2)

Unter Zuhilfenahme eines Routers mit IGMP-Snooping sind die Switche in der Lage ein multicast-ähnliches Verhalten zu ermöglichen.

MMRP hingegen ermöglicht ein Multicast auf Layer-2-Ebene, weswegen es ohne IGMP-Snooping funktioniert, also auch ohne zusätzlichen Router. Einzige Vorraussetzung ist die Verwendung von MMRP-fähigen Switchen.

## Modell des Problembereiches

Der aktuelle Multicastor kann zum Testen von wie folgt aufgebauter Netzwerken genutzt werden:

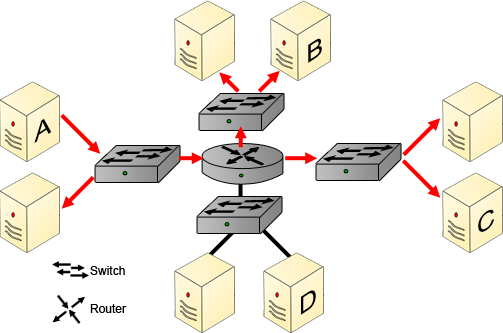


Abbildung : Netzwerkaufbau mit Router[[3]](#footnote-3)

Es ist zu erkennen, dass zwingend ein Router erforderlich ist. Ein rein auf Switchen basierendes Netzwerk wird noch nicht unterstützt. Aus diesem Grund wird in Version 2.0 des MultiCastors das MMR-Protokoll implementiert werden, um auch Netzwerke folgender Struktur testen zu können:

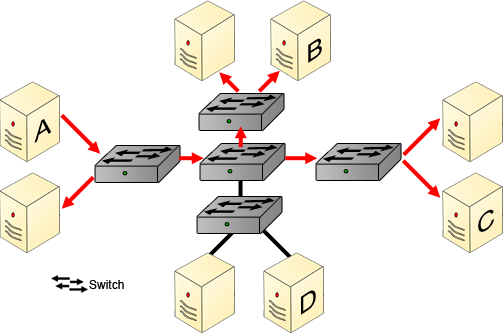


Abbildung 5: Netzwerkaufbau ohne Router

Generell ist der MultiCastor ein Tool, das das Netzwerk im Unternehmen auf Belastbarkeit und Multicast-Fähigkeit testen soll. Beim Testen kann es gegebenenfalls zu folgenden Komplikationen kommen:

1. Der Heap kann überlaufen, wenn zu viele Multicast-Ströme zeitgleich getestet werden.
2. Es könnte ein Verbindungsabbruch im Netzwerk auftreten.

Für beide Fälle werden Problemerkennung sowie -lösung implementiert werden:

Der MultiCastor V2.0 wird bei der Anlegung einer neuen Multicast-Instanz prüfen, ob genug Speicher zur Verfügung steht und anderenfalls eine entsprechende Warnung ausgeben.

Wenn ein Netzwerkabriss auftreten sollte, wird dieses Ereignis protokolliert, der User informiert und versucht, die Verbindung wieder herzustellen.

## Beschreibung der Geschäftsprozesse

Der MultiCastor unterstützt den Nutzer bei einer Vielzahl von Vorgängen beim Netzwerktest. Zur Verdeutlichung wird ein beispielhafter Testprozess in Abbildung 6 dargestellt.[[4]](#footnote-4)

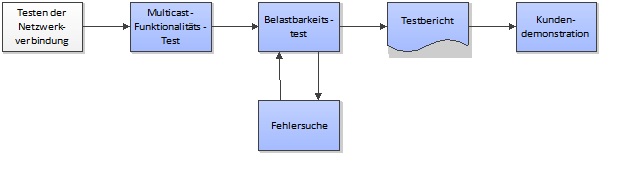


Abbildung : Beispiel Testprozess[[5]](#footnote-5)

Hierbei kann der Multicastor den Nutzer bei jeder blau markierten Tätigkeit unterstützen. Im Folgenden werden die angenommenen Aktivitäten genauer erklärt:

### Testen der Netzwerkverbindung

Hierbei wird sichergestellt, dass das netzwerkfähige Gerät auch mit dem Netzwerk verbunden ist und eine Verbindung zu einem anderen, im Netzwerk befindlichen Gerät herstellen kann, zum Beispiel über eine Ping-Anfrage.

### Multicast-Funktionalitätstest

Soll später das Multicast-Protokoll im Netzwerk verwendet werden, ist ein Test ratsam. Dieser stellt sicher, dass alle beteiligten Geräte Multicast unterstützen und das Netz fehlerfrei funktioniert. Hierbei ist der MultiCastor behilflich, indem Multicast-Datenströme von/an eine Vielzahl verschiedener Multicast-Gruppen gesendet und empfangen werden können. Darüber hinaus bietet es eine Jitter- und Lost Packets-Anzeige, die eine Fehlersuche ermöglicht.

### Belastbarkeitstest

Bei erwartetem, hohem Datenaufkommen im Netzwerk ist ein Belastbarkeitstest notwendig. Dieser kann durchgeführt werden, indem die Packets per Second innerhalb des Multicast-Servers sehr hoch gestellt werden, sodass das Netzwerk an seine Grenzen kommt.

### Fehlersuche

Durch die Übertragung einer SenderID und der Auswertung, wie viele Pakete verloren gegangen sind, lassen sich Schwachstellen im Netzwerk leicht aufdecken.

### Testbericht

Der MultiCastor erstellt eine Logdatei mit den gesammelten Testergebnissen, welche anschließend in einem Testbericht genutzt werden können.

### Kundendemonstration

Mit dem MultiCastor kann die Belastbarkeit eines Netzwerks direkt gezeigt werden. Eine graphische Darstellung der Messwerte ermöglicht hierbei eine optisch ansprechende Darstellung der Ergebnisse.

# Produktfunktionen

## Use Cases – Übersicht

Das folgende Diagramm zeigt die Abhängigkeiten der Use Cases. Als neue Rolle ist der Developer/Tester aufgeführt, der über STAF/STAX die Anwendung prüfen kann. Ebenfalls ist die Multiinstanzfähigkeit abgebildet. Weitere Erweiterungen finden sich bei den Konfigurationsmöglichkeiten.

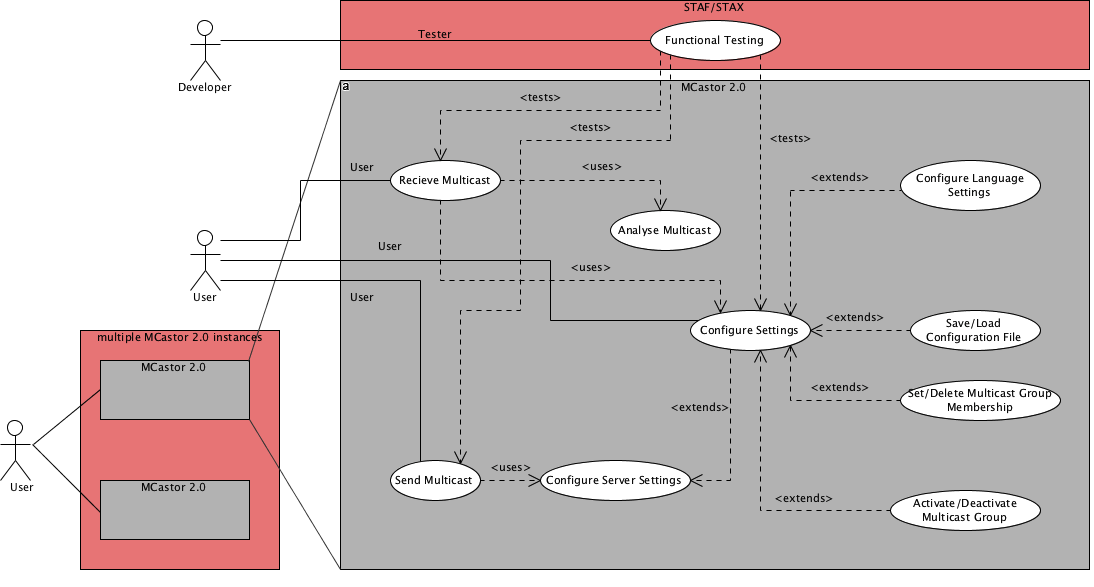


Abbildung : Übersicht Use-Cases

In den folgenden Abschnitten werden die einzelnen Use-Cases näher beschrieben.

## /LUC10/: Send Multicast

**Charakterisierende Informationen:**

|  |  |
| --- | --- |
| Ziel des Use Cases: | Dem User zu ermöglichen einen Multicast Stream anzulegen. Dieser Stream soll auf Layer 2 und Layer 3 Ebene kommunizieren. |
| Umgebende Systemgrenze: | Es wird auf die Netzwerkkarte des Computers zugegriffen. |
| Vorbedingung: | Es muss eine lauffähige Installation des MultiCastors2.0 auf dem Usersystem vorhanden sein.  Für das Senden von Multicast Streams sollte ein Betriebssystem mit IGMP und MLD Unterstützung vorhanden sein.  Für den MMRP Multicast Stream kann ein Switch mit MMRP Unterstützung genutzt werden. Dies ist jedoch nicht notwendig. |
| Nachbedingung bei erfolgreicher Ausführung: | Programm sendet Pakete. |
| Beteiligte Nutzer: | Jeder Anwender |

**Szenario für den Standartablauf:**

Der Benutzer kann auf der Startseite ein Sendertab zum Programm hinzufügen. Er sucht sich aus, ob ein Layer-2 oder -3 Multicaststrom betrieben werden soll. Es werden Standardwerte für den Sender geladen, so dass der User sofort einen Strom anlegen kann. Änderungen für die Verbindung kann können ebenfalls in diesem Tab vorgenommen werden.

## /LUC20/: Receive Multicast

**Charakterisierende Informationen:**

|  |  |
| --- | --- |
| Ziel des Use Cases: | Dem User zu ermöglichen einen Multicast Stream zu empfangen. Dieser Stream soll auf Layer-2- und Layer-3-Ebene erreichbar sein. |
| Umgebende Systemgrenze: | Es wird auf die Netzwerkkarte des Computers zugegriffen. |
| Vorbedingung: | Es muss eine lauffähige Installation des MultiCastors2.0 auf dem Usersystem vorhanden sein.  Für das Empfangen von Multicast Streams sollte ein Betriebssystem mit IGMP und MLD Unterstützung vorhanden sein.  Für den MMRP Multicast Stream kann ein Switch mit MMRP Unterstützung genutzt werden. Dies ist jedoch nicht notwendig. |
| Nachbedingung bei erfolgreicher Ausführung: | Programm sendet Pakete. |
| Beteiligte Nutzer: | Jeder Anwender |

**Szenario für den Standartablauf:**

Der Benutzer kann auf der Startseite ein Receivertab zum Programm hinzufügen. Er sucht sich aus, ob ein Layer-2 oder -3 Multicaststrom empfangen werden soll. Es werden Standardwerte für den Empfänger geladen, so dass der User sofort einen Strom empfangen kann. Änderungen für die Verbindung kann der User ebenfalls in diesem Tab vornehmen.

## /LUC30/: Analyse Multicast

**Charakterisierende Informationen:**

|  |  |
| --- | --- |
| Ziel des Use Cases: | Das Analysieren von Multicastströmen |
| Umgebende Systemgrenze: | Es wird auf die Netzwerkkarte des Computers zugegriffen. |
| Vorbedingung: | Ein Sender und ein Receiver müssen angelegt worden sein. |
| Nachbedingung bei erfolgreicher Ausführung: | Graph zeigt erhaltene Pakete an. |
| Beteiligte Nutzer: | Jeder Anwender |

**Szenario für den Standartablauf:**

Abhängig davon, ob man sich im Receiver- oder Sendertab befinden, zeigt der Graph dementsprechend die Werte an. Im Receivertab wird dargestellt, wieviele Pakete man bekommen hat bzw. verloren gegangen sind. Im Sendertab wird dargestellt, wieviele Pakete versendet wurden.

## /LUC40/: Configure Settings

**Charakterisierende Informationen:**

|  |  |
| --- | --- |
| Ziel des Use Cases: | Die Einstellungen zu Sender und Receiver können vom Benutzer vorgenommen werden. |
| Umgebende Systemgrenze: | MultiCastor 2.0 / GUI oder CMD mit Config-Dateien |
| Vorbedingung: | Es muss eine lauffähige Installation MultiCastor2.0 auf dem Usersystem vorhanden sein. |
| Nachbedingung bei erfolgreicher Ausführung: | keine |
| Beteiligte Nutzer: | Jeder Anwender |

**Szenario für den Standartablauf:**

Der Benutzer öffnet ein Sender/Receiver (Layer2 oder Layer3) Tab und will darin die Einstellungen für einen bestehenden oder neuen Multicast-Strom eintragen. Wenn bei einem bestehenden Multicast-Strom die Einstellungen geändert werden sollen, muss dieser zuerst in der Tabelle ausgewählt werden, dann werden die Einstellungen in das “*MulticastConfig Panel*” geladen. Wenn ein neuer Multicast-Strom angelegt werden soll, muss auf den “*Neu*” Button geklickt werden, sofern bereits ein Sender/Receiver gewählt ist.

Unter den UseCase Configure Settings fallen die gemeinsamen Einstellungen von Sender und Receiver. Diese sind in folgender Tabelle grün markiert.

|  |  |  |
| --- | --- | --- |
| Layer-3 | Layer-2 | Auch beim Receiver enthalten? |
| IP Group Address | MAC-Group-Address | y/y |
| Netwrk Interface | Network Interface | y/y |
| Packet Rate | Packet Rate | n |
| Packet Length | Packet Length | n |
| UDP Port | - | n |
| Time to Live | - | n |

Die “Group Address” hat unterschiedliche Wertebereiche, je nachdem, ob es um Layer2 oder Layer3(IPv4 oder IPv6) handelt. Diese Wertebereiche werden über die Protokolle spezifiziert:

|  |  |
| --- | --- |
| Typ | Zulässiger Wertebereich |
| IPv4 | 224.0.0.0 - 239.255.255.255 |
| IPv6 | Jede Adresse die mit FF00::/8 beginnt |
| MMRP | 80:00:00:00:00:00-ff:ff:ff:ff:ff:fe  Das **höchstwertigste Bit** der MAC-Adresse gibt an, ob es sich um ein Uni- oder Multicast-Adresse handelt.  0 => Quelle ist “Individual”, also Unicast  1 => Quelle ist “Group” also Multicast. Allerdings fällt hierrunter auch der Broadcast. Also das Senden von einem Sender an Alle(!) Empfänger im LAN. Hierfür ist normalerweise die Adresseff:ff:ff:ff:ff:ff reserviert. |

Für das *Network Interface* wird eine Auswahlbox mit den möglichen Network-Interfaces angezeigt, aus denen ein Interface ausgewählt werden muss. Hier ist jede Auswahl richtig, da nur benutzbare Interfaces angezeigt werden.

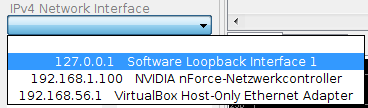


Abbildung : Auswahl Netzwerk Interface

### /LUC41/: Save Configuration File

**Charakterisierende Informationen:**

|  |  |
| --- | --- |
| Ziel des Use Cases: | Die Benutzereinstellungen (angelegte Sender/Receiver) werden in einem Configuration File gespeichert. Es soll auch ein partielles Speichern einzelner Teilbereiche möglich sein. |
| Umgebende Systemgrenze: | Es wird von dem MultiCastor2.0 in das lokale Dateisystem eine Datei geschrieben. |
| Vorbedingung: | Es muss eine lauffähige Installation des MultiCastors2.0 auf dem Usersystem vorhanden sein. |
| Nachbedingung bei erfolgreicher Ausführung: | keine |
| Beteiligte Nutzer: | Jeder Anwender |

**Szenario für den Standartablauf:**

Dem Benutzer wird das Speichern seiner aktuellen Konfigurationen über den Menüpunkt “Save Configuration” im Kontextmenüpunkt “Menu” ermöglicht.

Wählt der Anwender diese Option, kann er zunächst den Ort und den Dateinamen in seinem lokalen Dateisystem auswählen unter dem die Einstellungen gespeichert werden sollen.

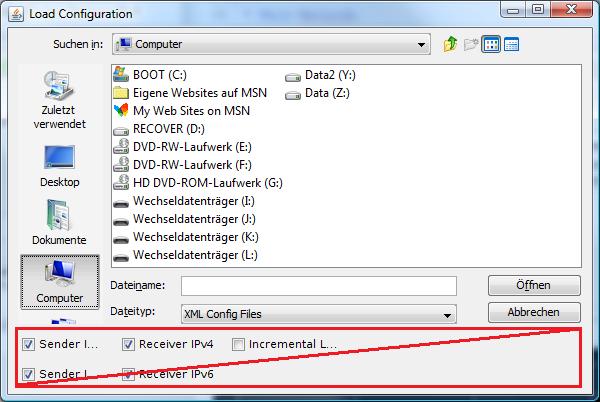


Abbildung : Dateiauswahlfenster mit hervorgehobenen Änderungen.

In diesem Dialog werden die Checkboxen zur partiellen Speicherung (unten) entfernt. Die Auswahloptionen für die partielle Speicherung werden stattdessen in ein eigenes Fenster ausgelagert, welches nach der Auswahl des Speicherziels angezeigt wird.

Dieses neue Fenster wird es ermöglichen, auch einzelne Receiver und Sender zu speichern oder von der Speicherung auszuschließen.

Um identifizieren zu können, wer zu welcher Zeit eine Konfigurationsdatei gespeichert hat, werden beim Speichern von Konfigurationen zukünftig die PC-Kennung und der Speicherzeitpunkt mit in die Datei geschrieben.

### /LUC42/: Load Configuration File

**Charakterisierende Informationen:**

|  |  |
| --- | --- |
| Ziel des Use Cases: | Der Benutzer lädt Sender- / Receivereinstellungen aus einem Configuration File in seinem lokalen Dateisystem. Es soll auch ein partielles Laden einzelner Teilbereiche möglich sein. |
| Umgebende Systemgrenze: | Es wird von dem MultiCastor2.0 von dem lokalem Dateisystem eine Datei gelesen. |
| Vorbedingung: | Es muss eine lauffähige Installation des MultiCastors2.0 auf dem Usersystem vorhanden sein. |
| Nachbedingung bei erfolgreicher Ausführung: | keine |
| Beteiligte Nutzer: | Jeder Anwender |
| Auslösendes Ereignis | Der Benutzer möchte gespeicherte Einstellungen (Sender/Receiver) laden. |

**Szenario für den Standartablauf:**

Dem Benutzer wird das Laden von auf seinem lokalen Dateisystem gespeicherten Konfigurationen für Sender und Receiver über den Menüpunkt “Load Configuration” im Kontextmenüpunkt “Menu” ermöglicht.

Äquivalent zum Speichervorgang (/LUC41/) wählt der Anwender zunächst eine Datei aus seinem Dateisystem aus, bevor er in einem zweiten Dialogfeld einzelne Sender oder Receiver vom Laden ausschließen oder zum Laden hinzufügen kann. Ein inkrementelles Laden mehrerer Dateien kann durch das Wiederholen dieses Vorgangs erreicht werden.

Hierbei wird auch /BUG50/ (Partielles Laden von Konfigurationen) behoben, da die fehlerverursachenden Kontrollkästchen in Version 2.0 nicht mehr benötigt werden.

### /LUC43/: Set/Delete Multicast Group Membership

**Charakterisierende Informationen:**

|  |  |
| --- | --- |
| Ziel des Use Cases: | Der Anwender soll einen Sender / Receiver zu einer Multicast-Gruppe hinzufügen, und sie auch wieder löschen können. |
| Umgebende Systemgrenze: | MultiCastor2.0 / GUI oder CLI mit Config-Dateien |
| Vorbedingung: | Es muss eine lauffähige Installation des MultiCastors2.0 auf dem Usersystem vorhanden sein. |
| Nachbedingung bei erfolgreicher Ausführung: | keine |
| Beteiligte Nutzer: | Jeder Anwender |
| Auslösendes Ereignis | Der Benutzer gibt einem Sender / Receiver eine Multicast-Gruppe |

**Szenario für den Standartablauf:**

Der Anwender kann unter “Multicast Configuration” eine Multicast Group Identifikation (Group IP oder “Mac-Multicast-Adresse”) angeben.

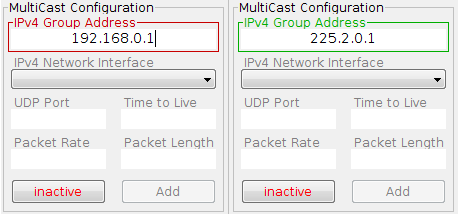


Abbildung : Eingabe einer Multicast-Adresse

Während der Eingabe wird überprüft, ob die Eingabe im jeweiligen Bereich liegt.

|  |  |
| --- | --- |
| Art | Beschreibung |
| IPv4 | 224.0.0.0 – 239.255.255.255 |
| IPv6 | Jede Adresse, die auf FF00::/8 beginnt |
| MMRP | 80:00:00:00:00:00 – ff:ff:ff:ff:ff:fe  Das **höchstwertigste Bit** der MAC-Adresse gibt an, ob es sich um ein Uni- oder Multicast handelt.  0 => Quelle ist „Individual“, als Unicast  1 => Quelle ist „Group“, also Multicast. Allerdings fällt hierunter auch der Broadcast, also das Senden von einem Sender an **alle** Empfänger im LAN. Hierfür ist normalerweise die Adresse ff:ff:ff:ff:ff:ff reserviert. |

Sobald die Eingabe einen zulässigen Wert hat, wird der Rahmen des Eingabefeldes von Rot auf Grün gestellt.

### /LUC44/: Activate/Deactivate Multicast Group

**Charakterisierende Informationen:**

|  |  |
| --- | --- |
| Ziel des Use Cases: | Der Anwender soll bei einem Sender/Receiver eine bereits hinzufügte Multicast-Gruppe aktivieren und sie wieder deaktivieren können. |
| Umgebende Systemgrenze: | MultiCastor2.0/GUI oder CLI mit Config-Dateien |
| Vorbedingung: | Es muss eine lauffähige Installation des MultiCastors2.0 auf dem Usersystem vorhanden sein. Außerdem muss ein Sender/Receiver mit Multicast-Gruppe existieren (/LUC43/) |
| Nachbedingung bei erfolgreicher Ausführung: | Das Senden oder Empfangen für den jeweiligen Receiver oder Sender wird eingestellt oder gestartet. |
| Beteiligte Nutzer: | Jeder Anwender |
| Auslösendes Ereignis | Der Benutzer aktiviert oder deaktiviert einen Multicast/Strom. |

**Szenario für den Standartablauf:**

Der Anwender kann die existierenden Sender oder Receiver aus dem “MultiCast Overview”-Panel auswählen. Ob dieser derzeit aktiv ist, ist daran erkennbar, ob der Haken in der “State” Spalte gesetzt ist.

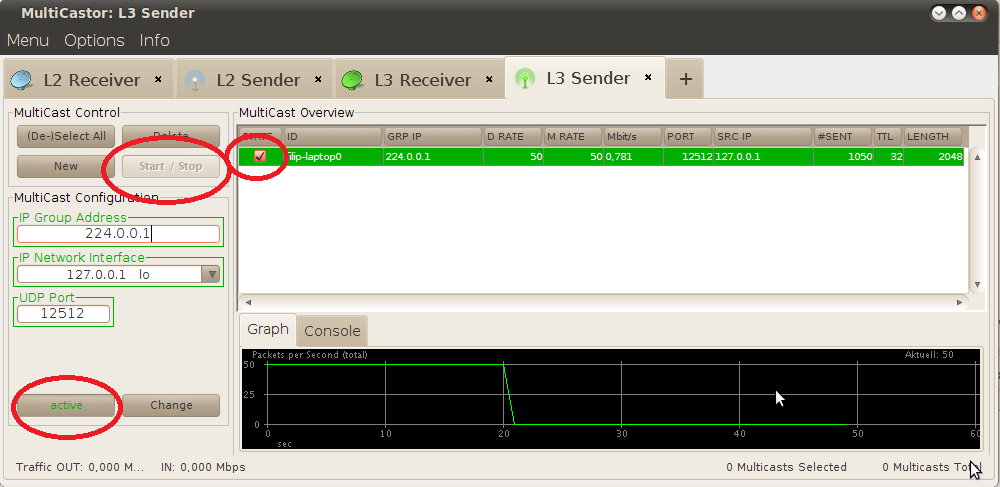


Abbildung : Aktivierung eines Multicast-Stromes

Nun kann der Anwender den Status ändern (von aktiv zu inaktiv oder umgekehrt), indem er die Checkbox in der “STATE”-Spalte der Tabelle aktiviert/deaktiviert oder bei ausgewähltem Sender bzw. Receiver auf den „Start / Stop“-Button drückt.

Außerdem kann der Sender/Receiver direkt beim Anlegen im “MultiCast Configuration” Panel über den Button active/inactive aktiviert werden.

Ein aktiver Sender versendet Multicast Daten und hat die zugehörige Multicast Gruppe registriert.

Ein aktiver Empfänger versucht Multicast Daten der zugehörigen Multicast Gruppe zu empfangen.

### /LUC45/: Configure Server Settings

**Charakterisierende Informationen:**

|  |  |
| --- | --- |
| Ziel des Use Cases: | Der Anwender soll einen Server (Sender) konfigurieren können. |
| Umgebende Systemgrenze: | MultiCastor2.0 / GUI |
| Vorbedingung: | Es muss eine lauffähige Installation des MultiCastors2.0 auf dem Usersystem vorhanden sein. |
| Nachbedingung bei erfolgreicher Ausführung: | keine |
| Beteiligte Nutzer: | Jeder Anwender |
| Auslösendes Ereignis | Der Anwender kann einen Server konfigurieren. Dies kann beim Erstellen oder beim nachträglichen Konfigurieren geschehen. |

**Szenario für den Standartablauf:**

Nachdem das Programm gestartet wurde, wählt der Nutzer den passenden Tab aus. Wenn Layer-2-Sender oder Layer-3-Sender ausgewählt wurden, kann der Benutzer die Daten für den Sender eingeben. Für Layer-3-Sender erscheint beispielsweise im unteren Bereich der GUI (roter Kasten) folgendes Einstellungspanel:

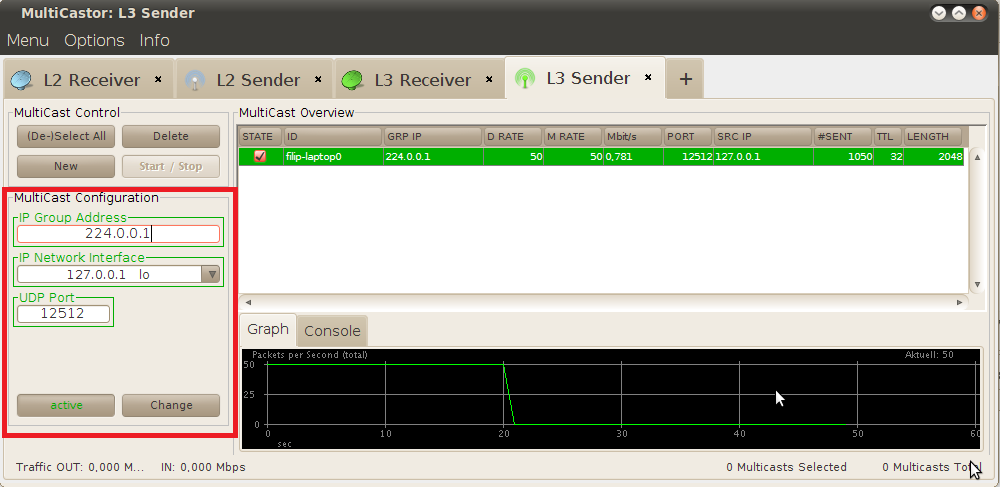


Abbildung : Anlegen eines Senders

Je nachdem, ob der Layer-2-Sender oder der Layer-3-Sender offen ist, werden hier verschiedene Werte erwartet:

|  |  |  |
| --- | --- | --- |
| Layer-3-Sender | Layer-2-Sender | Auch beim Receiver enthalten? |
| IP Group Address | MAC-Group-Address | y/y |
| IP Network Interface | Network Interface | y/y |
| UDP PORT | - | n |
| Time to Live | - | n |
| Packet Rate | Packet Rate | n |
| Packet Length | Packet Length | n |

Die Werte, die sowohl beim Receiver, als auch beim Sender gefordert sind, nämlich IP Group Address/IP Network Interface und MAC-Group-Address, fallen unter die allgemeinen Settings (/LUC40/ Configure Settings). Dieser Use Case erweitert die Daten um die nur beim Sender benötigten Informationen. Für die Daten gelten folgende zulässigen Werte:

|  |  |
| --- | --- |
| Typ | Zulässiger Wertebereich |
| Packet Length (byte) | Ganze Zahl von 52 bis 65.507 (IPv4)  Ganze Zahl von 52 bis 65.527 (IPv6)  Ganze Zahl von 64 bis *1518 (ohne VLAN-Tag)*(MMRP) |
| UDP Port | Ganze Zahl von 0 bis 65.635 |
| Time To Live | Ganze Zahl von 0 bis 255 |
| Packet Rate (pps) | Beliebige ganze Zahl |

### /LUC46/: Configure Language Settings

**Charakterisierende Informationen:**

|  |  |
| --- | --- |
| Ziel des Use Cases: | Der Anwender soll die Sprache wechseln können, indem er unterschiedliche Languagefiles auswählt. |
| Umgebende Systemgrenze: | MultiCastor 2.0 / Languagefile auf lokalem Dateisystem. |
| Vorbedingung: | Es muss eine lauffähige Installation des MultiCastors2.0 auf dem Usersystem vorhanden sein. |
| Nachbedingung bei erfolgreicher Ausführung: | keine |
| Beteiligte Nutzer: | Jeder Anwender |
| Auslösendes Ereignis | Der Benutzer wechselt die Sprache der Software |

**Szenario für den Standartablauf:**

Der Anwender kann die Sprache über den Kontextmenüpunkt “Optionen” bzw. den dortigen Unterpunkt “Sprache ändern” wählen. Zur Auswahl der Sprache dient ein dem folgenden Dialog ähnliches Fenster



Abbildung : Beispiel Sprachauswahl

Die Sprachdateien werden in einem fest definierten Verzeichnis abgelegt und mit der Sprachbezeichnung als Dateiname benannt.

Wird eine neue Sprache ausgewählt, kann dies einen Neustart des Programms oder von Teilen des Programms erforderlich machen, um Beschriftungen neu laden zu können. Diese Reloads werden, sofern möglich, automatisch vollzogen.

## /LUC50/: Multi-Instance-Ability

**Charakterisierende Informationen:**

|  |  |
| --- | --- |
| Ziel des Use Cases: | Es sollen mehrere Programminstanzen gleichzeitig gestartet werden können. Diese Instanzen sollen über die Loopback-Funktion (Windows 32 bit & Linux) auch lokal untereinander kommunizieren können.  Des Weiteren werden die Instanzen durch unterschiedliche Fenstertitel unterscheidbar gemacht. |
| Umgebende Systemgrenze: | Es handelt sich um mehrere Instanzen des MultiCastors2.0, die über die Ethernetschnittstellen miteinander kommunizieren können. |
| Vorbedingung: | Es muss eine lauffähige Installation des MultiCastors2.0 auf dem Usersystem vorhanden sein. |
| Nachbedingung bei erfolgreicher Ausführung: | keine |
| Beteiligte Nutzer: | Jeder Anwender |
| Auslösendes Ereignis | User verwendet mehrere Programminstanzen gleichzeitig. |

**Szenario für den Standartablauf:**

Der Anwender öffnet mehrere Programminstanzen zur gleichen Zeit auf seinem System. In jeder Programminstanz stehen nun die Funktionen zum Senden und Empfangen zur Verfügung. Somit können die geöffneten MultiCastor2.0 sowohl untereinander kommunizieren (Loopback), als auch an externe Ziele Daten senden oder von externen Sendern Daten empfangen.

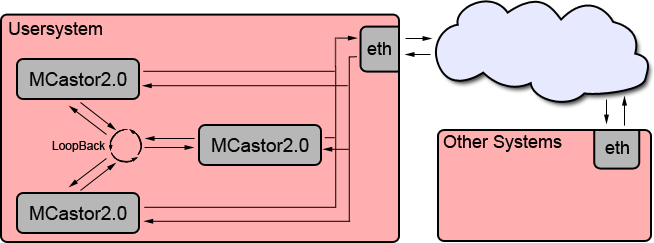


Abbildung : Kommunikation mehrere MC2.0 Instanzen

**Usability Verbesserungen für Multiinstanznutzung**

**Konfigurierbare Instanztitel:**

Das Fenstermanagement der MultiCastor2.0 wird dahingehend erweitert, dass mehrere Instanzen anhand eines unterschiedlichen Fenstertitels unterscheidbar sind. Der Fenstertitel wird sich aus einem frei wählbaren und optionalen ersten Abschnitt und der Bezeichnung des aktuell aktiven Tabs zusammen setzen.

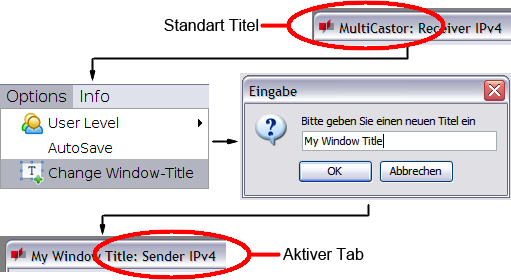


Abbildung : Workflow bei der Änderung des Fenstertitels

**Ausblenden und Verschieben von Tabs:**

Um, über die Unterscheidbarkeit von Instanzen hinaus, die Navigation in den einzelnen Programmfenstern zu erleichtern, werden die Sender- und Receivertabs ausblendbar gestaltet. Darüber hinaus werden die Tabs verschiebbar sein und können somit nach den persönlichen Präferenzen des jeweiligen Benutzers angeordnet werden.

|  |  |
| --- | --- |
| https://lh4.googleusercontent.com/yc3K1Scur5Fv4zf5Yv4Kd-tiUgMj1n9BnoHMo-MV1hiY58EtSTb0jIyiGDmexcbcasUOCMKfduOnW_AJ29WghgwOLZqhdoL8kfx7NJXhmT16OxAElhE  Abbildung : Verschieben von Tabs | https://lh3.googleusercontent.com/nRe51XlfYEB0xtacYv9XgyqWr9gvZ8XQQlBbEr4Wpl-_LiFpFJUjRVm3wFJ7ECAAINsStQpy5JGeTXCMci8EVYUOgesGrTNTX4JaCJ-kx1SQgLjGY18  Abbildung : Öffnen und Schließen von Tabs |

Die optische Gestaltung der Tabs und die funktionale Umsetzung der Einblendung neuer Tabs wird der Gestaltung der Tabs im Browser eines bekannten Internetsuchanbieters nachempfunden, sodass die meisten Anwender mit der Bedienung bereits vertraut sind.

https://lh5.googleusercontent.com/Bbo9f9nT8WhcC0vN0xc_CShmUAMtPzykddSmrChrh5oRHLw8S6HKBrG5fQv58sdq4KL5Qg2UFHRV5A82Gasv1SuEaJujPRjMDhYAyqo3-3RIjYQiLz4

Abbildung : Darstellung der Tabs

## /LUC60/: Functional Testing

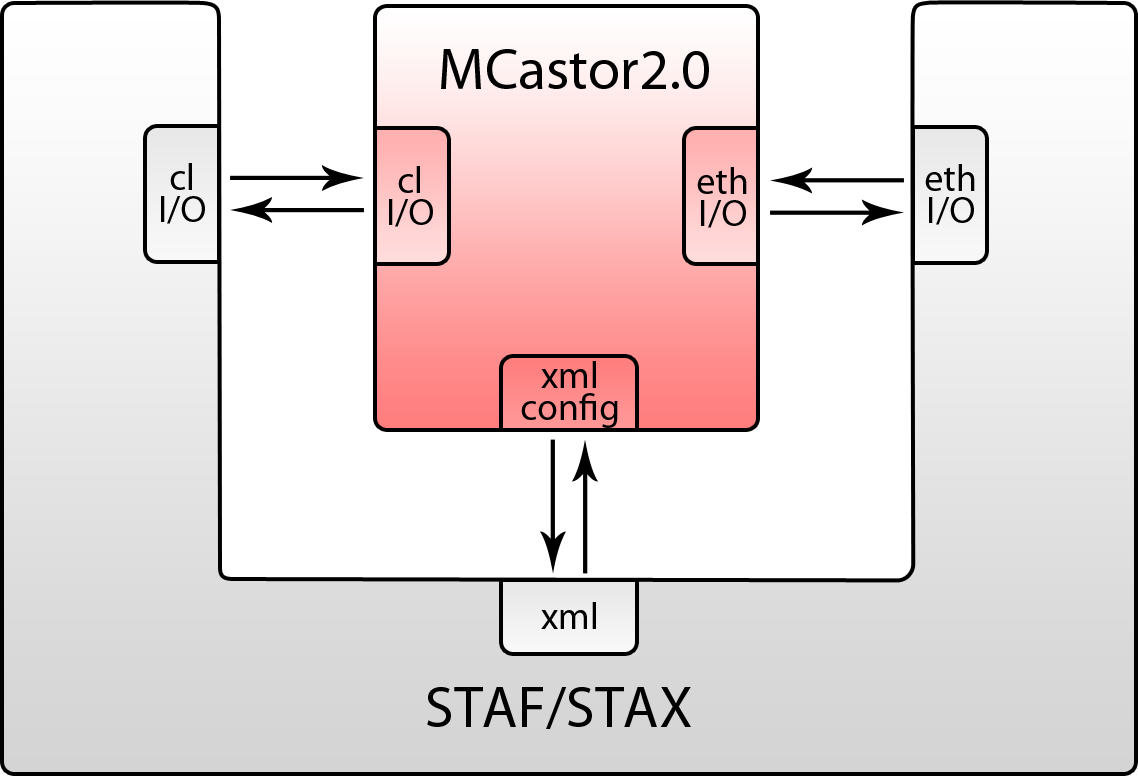


Abbildung : Functional Testing: Zugriffsmöglichkeiten mit STAF/STAX

**Charakterisierende Informationen:**

|  |  |
| --- | --- |
| Ziel des Use Cases: | Developer und Tester sollen mit STAF/STAX die XML-Konfigurationsdatei, die Korrektheit der Ethernetkommunikation und die Programmfunktionen (Zugriff über CLI) prüfen können. |
| Umgebende Systemgrenze: | Das Testobjekt ist der MultiCastor2.0. Als externe Testumgebung wird STAF/STAX verwendet.  https://lh6.googleusercontent.com/ABG9x9TT2cLkUVZgHvVY-dcGOE6zqWyd-y36snSDSRsTqk7YuNyAyh42TlY1Wi9_RXp3AmxIjCD9tUApH7ciWngj_ovAlICH51jm6DJ1XuLtJPClsZg  Abbildung : Systemgrenzen STAF/STAX & MC2.0 |
| Vorbedingung: | Es müssen lauffähige Installationen von STAF/STAX und MultiCastor2.0 auf dem Usersystem vorhanden sein. |
| Nachbedingung bei erfolgreicher Ausführung: | keine |
| Beteiligte Nutzer: | Developer, Tester und Administratoren, die die Programmfunktion testen möchten.  Der User kommuniziert mit STAF/STAX, welches wiederum mit dem MultiCastor2.0 kommuniziert. |
| Auslösendes Ereignis | User testet Anwendung mit STAF/STAX |

**Szenario für den Standartablauf (Erfolg):**

STAF/STAX ist eine externe Anwendung mit der die korrekte Funktionalität des MultiCastors2.0 getestet werden kann.

Hierfür führt der Developer/Tester verschiedene STAF/STAX Skripte aus, die die Programmfunktionalität, von verschiedenen Ausgangspunkten aus, prüfen.

Werden die Skripte im Testablauf fehlerfrei prozessiert, wird eine Erfolgsmeldung ausgegeben und es kann davon ausgegangen werden, dass die Anwendung in der getesteten Funktionalität fehlerfrei ist.

Zur Verdeutlichung des Ablaufs zeigt die Grafik eine beispielhafte Kommunikation zwischen einem STAF Server und einem Client, bei der ein date-Command ausgeführt wird.

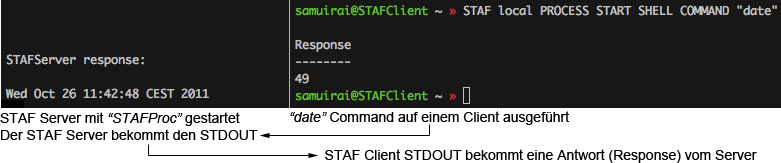


Abbildung : Beispielkommunikation zwischen STAF-Server und Client

**Szenario für den Fehlerfall (Programmfehler gefunden):**

Anhand der gewählten Ausgangspunkte für die einzelnen Testskripte ist eine Gliederung erfolgt. Im Fehlerfall ist identifizierbar, bei welchem Skript ein Fehler aufgetreten ist. Hieraus lassen sich Rückschlüsse auf die Fehlerstelle im Programm ziehen.

Das Team wird eine Auswahl an Beispielskripten zum Test einzelner Funktionen bereitstellen, die dem Developer/Tester als Vorlage dienen sollen, wie er mit eigenen Skripten verschiedene Funktionalitäten testen kann.

# Produktcharakteristiken

## Systemumgebung

Um immer volle Funktionalität zu garantieren, stellt der MultiCastor einige Mindest-Anforderungen an die Hardware- und Softwareumgebung. Im Folgenden werden die genauen Abhängigkeiten und Voraussetzungen aufgelistet.

Das Multicastor-Tool läuft sowohl auf Windows als auch auf unixbasierten Betriebssystemen. Das System muss über eine intakte Netzwerkverbindung oder eine Loopback-Funktion verfügen, um Tests durchzuführen.

## Hardwareumgebung

Die Hardware muss so gewählt werden, dass eine Java-Laufzeitumgebung auf Windows oder einem unixbasierten Betriebssystem installiert und dauerhaft betrieben werden kann. Die Hardware muss über eine Netzwerkschnittstelle verfügen, auf die über eine Softwareschnittstelle zugegriffen werden kann.

Für Multicast-Ströme, die das MMR-Protokoll benutzen wird außerdem ein MMRP-fähiger Switch benötigt, um MMRP-Anfragen verarbeiten zu können.

## Softwareumgebung

Der MultiCastor wurde und wird komplett in der Programmiersprache Java V1.6 umgesetzt. Es werden bei der Weiterentwicklung des MultiCastors die Standard-Java Bibliotheken sowie die JPCAP-Bibliothek[[6]](#footnote-6) verwendet. Letztere wird mit dem Produkt ausgeliefert, demnach muss nur eine JRE der Version 1.6 auf dem Hostsystem installiert und verfügbar sein.[[7]](#footnote-7)

# Anhang

## Multiple MAC Registration Protocol

Das MMR-Protokoll wurde 2007 von der IEEE 802.1ak spezifiziert. Das Protokoll ist ein Layer-2-Protokoll und ermöglicht es dadurch Switches, Multicasting zu betreiben.

Multicast erlaubt dem Sender seine Information an mehrere Empfänger im Netzwerk zu verschicken und das alles beim einmaligen Versenden. Die Pakete werden im Switch vervielfältigt und an die Empfänger versendet.

Der Sender eröffnet eine Multicast-Gruppe und wenn ein Empfänger Daten von dem Sender erhalten will, kann er sich bei der Gruppe registrieren und die gewünschten Inhalte erhalten. Selbstverständlich kann der Empfänger jeder Zeit die Verbindung trennen.

Die Vorteile von Multicasting, insbesondere mit dem MMR-Protokoll, sind:

* Netzwerk wird entlastet, da der Sender nur einmal seine Daten senden muss.
* Ein Empfänger erhält Daten nur, wenn er sie auch erhalten möchte.
* Ohne MMRP (Layer-2) muss sich zwingend ein Router im Netzwerk befinden, der die (Layer-3-)Pakete routet.

Um Multicasting mit dem MMR-Protokoll zu betreiben, benötigt man Switche, die dieses Protokoll unterstützen. Zurzeit sind nur wenige Switche auf dem Markt, die MMRP unterstützen.

## Testframework STAF/STAX

An dieser Stelle soll ein kleiner Überblick über STAF/STAX gegeben werden. Es wird erläutert, was es macht und wie es Aufzusetzen ist.

### STAF

STAF ist ein Testautomations-Framework, mit dem man komfortabel multiple Tests auf hunderten von Rechnern ausführen kann. Dafür liefert STAF viele verschiedene Services, die zum Testen benutzt werden können.

**Grundlegendes:**

*“STAFProc”* startet einen STAF Server Prozess

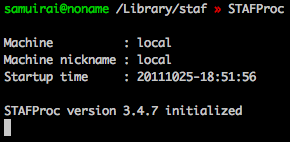


Abbildung : STAF starten

STAF mit dem Befehl *“STAF local shutdown shutdown”* aus einer anderen Terminal Session beenden. Bevorzugt hiermit, um nicht mit Ctrl+C eventuell Inkonsistenz zu erhalten. Ab sofort werden wie in Abb. 22 gezeigt, links der STAF Server laufen und im rechten Client Befehle abgegeben.

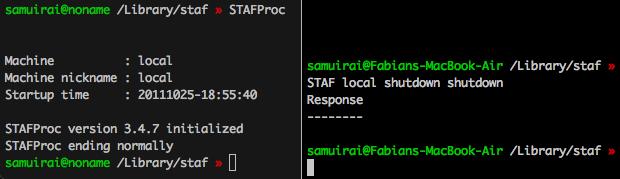


Abbildung : STAF beenden

**STAF Befehle:**

STAF <Endpoint> <Server> <Request>  
*<Endpoint>* “*local”* wenn man die STAF Befehle lokal ausführen will, oder einen anderen PC Namen, wenn auf einem Remote Computer.  
*<Service>* Name des Services der ausgeführt werden soll.  
*<Request>* Ist der Service Request  
*“The STAF command line utility works just like any other STAF application. It registers with STAF, performs a request (which is the service request you specify), and then unregisters. That last step causes the handle to be deleted. This somewhat limits the usage of the STAF command line utility.“[[8]](#footnote-8)*

**STAF Demo[[9]](#footnote-9):**

Bei einer STAF Demo, aus einer der STAF Dokumentationen, kann man sehen, wie mächtig STAF ist. Dort wurde ein eigener STAF Service in Java geschrieben, der die Ausführung von einem simplen Java Programm übernimmt. *(Bei der Kommandozeile wird ein Handler immer wieder beendet. Bei einem eigenen Service kann man die Verbindung aufrecht halten)*. Hier kommt dann auch STAX ins Spiel - Die STAF Excecution Engine übernimmt die Aufgabe des Services schreiben und bietet eine einfache, freie, aber mächtige Skriptsprache die auf XML und Python aufbaut.

## STAX

STAX (STAF eXecution Engine) ist ein STAF Service, der es mit XML und Jython ermöglicht Tests zu definieren und diese über den STAF Monitor auszuführen.

# Dokumentversionen

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **>** |  |  |  |  | **18.10.2011 Filip Haase:**  Fachbegriffe Glossar: Dokument angelegt |
|  |  | **>** |  |  |  | **19.10.2011 Jonas Traub:**  SRS Abschnitt 3 (Produktfunktionen): Dokument angelegt und Gliederung erstellt |
|  |  | **>** |  |  |  | **21.10.2011 Jonas Traub:**  Übersichtsgrafik & Beschreibung ergänzt Charakteristische Informationen zu /LUC60/ Charakteristische Informationen zu /LUC50/ |
|  | **>** |  |  |  |  | **25.10.2011 Fabian Fäßler:**  STAF/STAX spezifische Begriffe hinzugefügt |
|  |  | **>** |  |  |  | **26.10.2011 Jonas Traub:**  Detailbeschreibung zu /LUC60/ Detailbeschreibung zu /LUC50/ Charakteristische Informationen zu /LUC41/ Charakteristische Informationen zu /LUC42/ Charakteristische Informationen zu /LUC46/ Detailbeschreibung zu /LUC46/ |
|  |  |  | **>** |  |  | **26.10.2011 Filip Haase:**  SRS Abschnitt 4 (Produktcharakteristiken): Dokument angelegt |
|  |  |  | **>** |  |  | **26.10.2011 Filip Haase:**  Inhalte zu System-, Hardware- und Softwareumgebung hinzugefügt |
|  | **>** |  |  |  |  | **26.10.2011 Filip Haase:**  Fachbegriffe zu Java hinzugefügt |
|  |  |  |  | **>** |  | **27.10.2011 Sebastian Koralewski:**  SRS Abschnitt 2 (Produkteinsatz): Dokument angelegt  Beschreibungen und Grafiken zu Abschnitten 2. bis 2.4 eingefügt |
|  |  | **>** |  |  |  | **27.10.2011 Filip Haase:**  Charakteristische Informationen zu /LUC43/ Detailbeschreibung zu /LUC43/ Charakteristische Informationen zu /LUC44/ Detailbeschreibung zu /LUC44/ Charakteristische Informationen zu /LUC45/ |
|  |  | **>** |  |  |  | **27.10.2011 Jonas Traub:**  Detailbeschreibung zu /LUC41/ Detailbeschreibung zu /LUC42/ |
|  | **>** |  |  |  |  | **27.10.2011 Jonas Traub:**  Abkürzungen aus SRS Abschnitt Produktfunktionen eingefügt |
|  |  | **>** |  |  |  | **27.10.2011 Sebastian Koralewski:**  Charakteristische Informationen zu /LUC10/ Charakteristische Informationen zu /LUC20/ |
|  | **>** |  |  |  |  | **28.10.2011 Jonas Traub:**  Fehlende Beschreibungen zu Abkürzungen hinzugefügt |
|  |  | **>** |  |  |  | **28.10.2011 Jonas Traub:**  Informationen zu Anforderungen unter /LF70/ in UC eingefügt |
|  |  |  |  |  | **>** | **29.10.2011 Matthis Hauschild**  SRS Abschnitt 1. Einleitung und Zielbestimmung erstellt |
| **>** |  |  |  |  |  | **MCastor2.0 Document Template** |
|  |  | **>** |  |  |  | **01.11.2011 Matthis Hauschild**  Dokumente zusammen geführt und in MCastor2.0 Template eingefügt und Fehlerkorrektur.  Neues Dokument: SRS Final v1.0 beta |
|  | **>** |  |  |  |  | **01.11.2011 Matthis Hauschild**  in MCastor2.0 Template eingefügt und Fehlerkorrektur |
|  |  | **>** |  |  |  | **01.11.2011 Matthis Hauschild**  Aktuelles Glossar, MMRP-Beschreibung und STAF/STAX-Informationen als Anlage angefügt |
| **>** |  |  |  |  |  | **03.11.2011 Sebastion Koralewski / Fabian Fäßler**  Ergänzende Informationen zu STAF/STAX und MMRP (Anlagen) erstellt und hunzugefügt. |
|  |  | **>** |  |  |  | **05.11.2011 Matthis Hauschild**  Änderungen von Filip Haase an /LUC40/ und /LUC45/ eingefügt |
|  |  | **>** |  |  |  | **06.11.2011 Matthis Hauschild / Jonas Traub**  Link zu JPCAP hinzugefügt und Formulierungen überarbeitet |
|  |  | **>** |  |  |  | **07.11.2011 Matthis Hauschild / Jonas Traub**  Verschiedenste Änderungen |
|  |  | **>** |  |  |  | **09.11.2011 Matthis Hauschild**  Verschiedenste Änderungen |
|  |  | **>** |  |  |  | **10.11.2011 Jonas Traub**  Dokument Finalisiert |

1. Quelle: http://images.tecchannel.de/images/tecchannel/bdb/383253/890.jpg [↑](#footnote-ref-1)
2. Quelle: http://www.h3c.com/portal/res/200706/01/20070601\_109047\_image002\_201356\_57\_0.gif [↑](#footnote-ref-2)
3. Quelle: http://www.bogdanturcanu.ro/wp-content/uploads/2010/07/igmpsnoop2.gif [↑](#footnote-ref-3)
4. Quelle: TIT09AiB\_Pflichtenheft\_MultiCastor\_1v0 Punkt 2.4 [↑](#footnote-ref-4)
5. Quelle: TIT09AIB\_Pflichtenheft\_MultiCastor\_1v0 [↑](#footnote-ref-5)
6. http://netresearch.ics.uci.edu/kfujii/Jpcap/doc/ [↑](#footnote-ref-6)
7. http://www.java.com/de/download/help/sysreq.xml [↑](#footnote-ref-7)
8. Quelle: <http://staf.sourceforge.net/current/STAFGS.pdf> 4.3 Submitting STAF Request from Command Line [↑](#footnote-ref-8)
9. <http://staf.sourceforge.net/current/STAFGS.pdf> 8. Getting started with STAF [↑](#footnote-ref-9)