D8.5 Combined Evaluation Report

Grant Agreement nr	780470
Project acronym	SAUCE
Project start date (duration)	January 1st 2018 (36 months)
Document due:	31/12/2020
Actual delivery date	31/12/2020
Leader	DNEG
Reply to	William Greenly - wmg@dneg.com
Document status	Submission Version

Project funded by H2020 from the European Commission

Project ref. no.	780470
Project acronym	SAUCE
Project full title	**S**mart **A**sset re-**U**se in **C**reative **E**nvironments
Document name	D8.5 - Combined Evaluation Report
Security (distribution level)	Public
Contractual date of delivery	31/12/2020
Actual date of delivery	31/12/2020
Deliverable name	Combined Evaluation Report
Type	Report
Status & version	Submission Version
Number of pages	38
WP / Task responsible	DNEG
Other contributors	-
Author(s)	William Greenly - DNEG
EC Project Officer	Ms Adelina Cornelia Dinu - adelina-cornelia.dinu@ec.europa.eu
Abstract	This document provides integrated and consolidated results for the evaluation plan detailed in D8.3. It combines all the partners evaluation in a single report.
Keywords	Search, User Interface, Prototype, API, keywords, properties, filters, tags, similarity, synonymity, crowd, animation, path, storage, asset management, test, evaluation
Sent to peer reviewer	Yes
Peer review completed	Yes
Circulated to partners	No
Read by partners	No
Mgt. Board approval	No

Document History

Version and date	Reason for Change
1.0 01-12-20	Document created by William Greenly
1.1 23-12-20	Version for peer review
1.2 30-12-20	Final version for submission to EC

Table of Contents

1 EXECUTIVE SUMMARY

This document provides a comprehensive and combined evaluation of a number of key work packages for the SAUCE project. It is the conclusion of the self assessment and evaluation plan detailed in D8.3 Evaluation Plan.

We begin by providing some background and context to the evaluation, in particular drawing the reader's attention to the extraordinary circumstances surrounding the project and associated deliverables.

We then proceed to detail and describe the tests and evaluation for the search and transformation framework, providing insights into the framework's usability and accuracy, along with areas for further testing and improvement.

Following on from this we then proceed to detail the evaluation plans for the work done as part of Asset Storage on Flix. We test both the improvements to its performance, capacity, scalability and availability.

One of the core characteristics of this project has been the continued and effective collaboration between partners along with a collective profession of the whole being greater than the sum of all the parts and this is exemplified in the interoperability and integration evaluation between the search and transformation framework, and the asset storage. We demonstrate and provide insight into how the two developments have been integrated together to complement each other and increase the contributions of both.

Finally, we move onto the crowd and path animation tests covering work packages 5 and 6. We evaluate the tools and techniques developed during the course of these work packages with real artists solving real world problems, highlighting areas where improvements and enhancements have been made possible.

2 BACKGROUND

This document provides the combined and consolidated results of all the evaluation tests planned in D8.3 Evaluation Plan. The final test results detailed in this document are consistent with this test plan with only some minor variations, notably:

- The tests in the original plan were grouped by consortia members. In the resulting report, it has been consolidated and presented as a unified result demonstrating the integration and collaboration between partners.
- As detailed in the test plan and prevalent in this deliverable, certain aspects of the evaluation have deviated over the course of the project due to the impact of COVID-19 on the partners, in particular DNEG. This has resulted in the combination of certain tests, reduction in testers and artists and a general reduction in the scope of the evaluation plan.

3 INTRODUCTION

This chapter provides an overview of the combined evaluation, re-establishes the goals and objectives of the combined evaluation and describes how we are able to demonstrate the capability of various deliverables and their constituent parts.

3.1 Main objectives and goals

- To provide a combined test and evaluation of the work packages detailed in section 3.4, based on the plan laid out in D8.3
- To provide further insights into additional testing and evaluation along with further development opportunities.

3.2 Methodology

The report consists of a number of evaluations conducted with real users, both expert and non-expert, dependent upon the subject of the evaluation. Evaluations consist of both quantitative and qualitative feedback and the evaluations provide summaries and conclusions of the evaluation along with excerpts from test diaries where appropriate.

3.3 Relationship to Self Assessment

This evaluation is the conclusion of the self assessment detailed in D1.2. As mentioned throughout this document, there are deviations to the original self assessment due to extraordinary circumstances.

3.4 Relationship to Other Work Packages

This deliverable evaluates the following work packages and deliverables:

- D4.4 Smart Search Prototype
- D5.4 Tools for Editing Mocap Data
- D5.5 Tools for Splicing Together Animation Clips
- D7.3 Prototype of Asset Store
- D6.8 Crowd scene synthesis and metrics for quality evaluation

It is the conclusion and result of D8.3 Evaluation Plan

4 Smart Search Prototype Test

Originally there were two tests intended for the Smart Search Prototype, but as detailed in D8.3, these were combined and reduced to just one test, 'Locating assets for a bid document'

This test involves a number of artists using the Smart Search Prototype to find assets for a bid document. This test is designed to measure two aspects of the user interface, first the usability and second the accuracy. The raw test results can be found here:

https://drive.google.com/file/d/1pajLmq3KKGlqXeQhzq4fujBoOS2cKQAL/view?usp=sharing

A summary of each test and associated results is provided below:

4.1 Usability

A simple usability test with two users was conducted to measure the time it takes a user to familiarise themselves with the user interface and to see how intuitive the controls are. The test required two *non-expert* (i.e in this case, none of the users had used or seen the user interface) users to complete a number of tasks and then measure the time it took. Below is the test script with the test results:

Task	**Average Time**
Search by a keyword	15 secs
Search by multiple keywords	15 secs
Search by keyword and a tag	2 mins
Search by multiple keywords and multiple tags	1 min
Filter by kind	1 min
Find property filters	1 min
Filter by property filter	1 min
View an asset	5 secs
Browser asset renderers	10 secs
Find similar assets	5 secs

The results are the average time taken to complete each task and the time to familiarise is included. In addition, individual feedback was captured and analysed.

Overall the familiarisation results and usability appear very favourable. It shows that in under 10 minutes a non-expert, first time user can familiarise themselves with the system and start to produce results. This is a considerably significant amount of time since the user interface could be used by thousands of artists across the globe, therefore the difference between 10 minutes and 1 hour is in the order of 1000 hours in total, a large amount of time and materials, particularly for an agency that depends on utility, efficiency and output.

4.2 Accuracy

In this instance a series of assets with associated criteria were lifted from a bid document and provided to testers. Testers were expected to find assets matching the criteria using the user interface and record the number of matches, the noise level (bad results), the time taken to find the matches and any other feedback. These were compared against the actual matches to determine accuracy. A summary of the results are provided below:

Criteria	Criteria	Type	Avg. Noise	Avg. Time	Avg Accuracy
airplane		Any	0%	2 mins	100.00%
tank		Any	0%	2 mins	33.00%
car		Any	0%	3 mins	65.00%
assembly line		Any	0%	2 mins	100.00%
elevator shaft		Any	0%	30 secs	100.00%
vapour trail		Any	100%	2 mins	100.00%
face mask		Any	33%	3 mins	100.00%
Angry crowd		Any	0%	2 mins	100.00%
Flying birds		Any	94%	2 mins	0.00%
Underground complex		Any	0%	2 mins	33.00%
Underground complex		Environment	0%	2 mins	100.00%
Hallway		Any	0%	2 mins	100.00%
Hallway		Environment	0%	3 mins	100.00%
Concrete		Any	0%	1 min	100.00%
Concrete		Image	0%	2 mins	100.00%
Concrete	Height 1024	Image	0%	3 mins	100.00%
Concrete		Texture	0%	30 secs	100.00%
brick		Any	0.00%	1 min	100.00%

brick		Image	10%	1 min	100.00%
brick	image/jpeg	Image	0%	2 mins	100.00%
brick	8 bit data precision	Texture	0%	2 mins	100.00%
Total			11.00%	1.9mins	87.00%

As we can see we have fairly consistent results. In some instances we had a complete failure for a user to find results when some were present and in some instances the user only found noise when there were no actual results. It seems that searches with some adjectives or adverbs cause the most issues, but by and large the results are positive, although the search times indicate some usability issues since the variance is quite high, even though the search criteria is quite uniform. More extensive testing might discover why.

4.3 Results

The initial results of both tests prove promising. It's clear that the user interface and user experience is intuitive and friendly. The search proves to be relatively accurate with some areas for improvement and areas where noise can be reduced. Most of the feedback presented from users was positive with some areas of confusion around keywords and tag usage, which led to some unusual times for certain searches. The filters and type navigator were useful and usable which is also positive.

Due to the COVID impact, the testing was far less extensive than originally intended, so it would be advisable to conduct another round of more thorough testing in specific areas. Additionally, a broader set of users with different abilities should also be engaged. Accessibility for a range of users was not addressed in this round of testing which would also prove insightful. But for a first pass the results were encouraging.

5 Flix Prototype Test

5.1 Asset Retrieval Time

This test was to assess how timely an asset can be searched, assessed for suitability and then downloaded into a new use case.

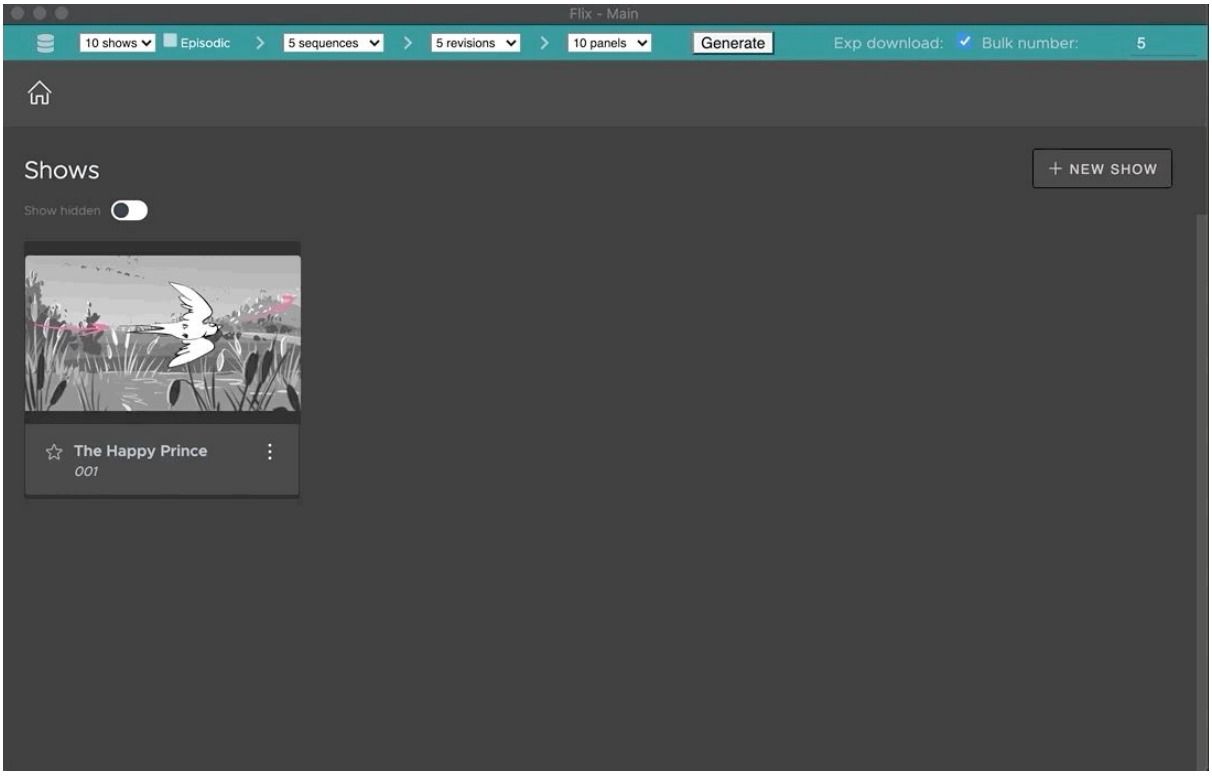
This test scenario is demonstrated with side-by-side comparisons of Flix 5 and Flix 6 which contains all the changes implemented in the SAUCE project. Utilising the UI of a sequence revision containing 100 panels.

5.1.1 Test details

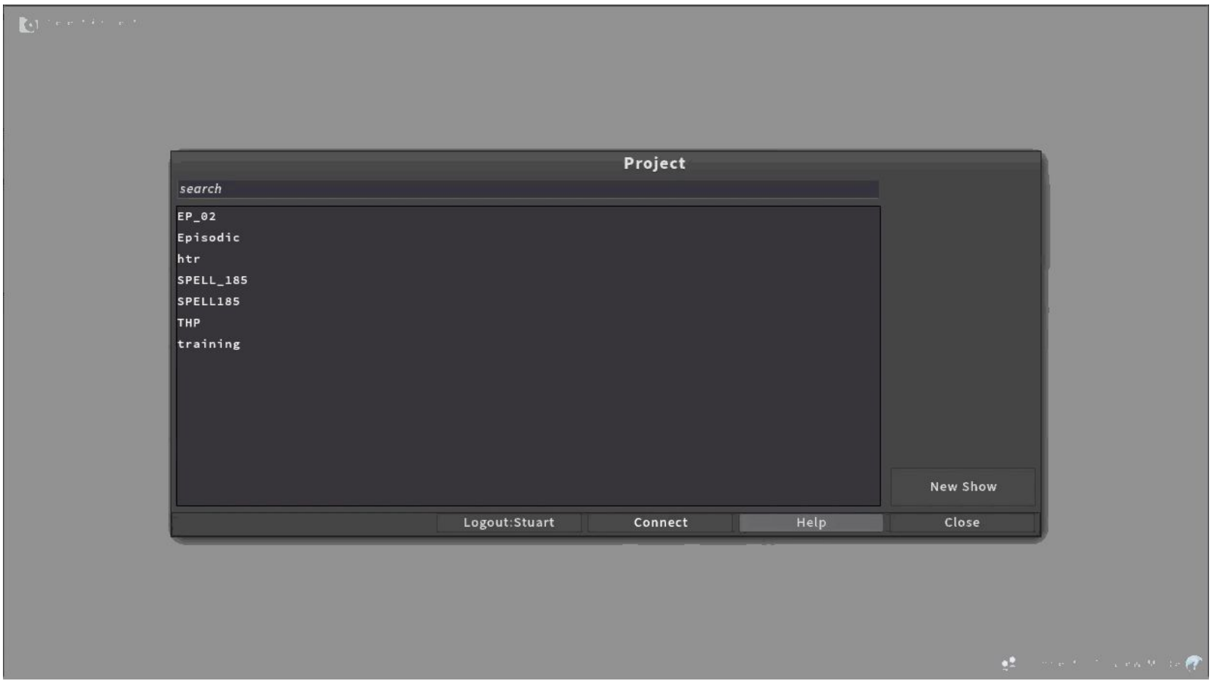
To carry out the test we asked 5 internal Flix users to follow a set of instructions. To remove discrepancies in build issues, all testing was carried out on Linux computers.

They first had to perform these tests in Flix 5, and then Flix 6. To avoid familiarity with the scene becoming a component factor, slightly different scenes and instructions were given for Flix 5 and Flix 6. Users were asked to time themselves carrying out the instructions, and were interviewed afterwards on their experience.

Flix 5 instructions:	Flix 6 instructions:
● Start the timer ● Find the “THP” show ● Find the “Rainy Nap Spot” Sequence ● Open the most recent sequence revision. ● Wait until the sequence is fully loaded. ● Stop the timer.	● Start the timer ● Find the “THP” show ● Find the “Rainy Nap Spot” Sequence ● Open the most recent sequence revision. ● Wait until the sequence is fully loaded. ● Stop the timer.



Flix 6 Example Test Scenario



Flix 5 Example Test Scenario

5.1.2 Results

The asset retrieval time evaluation we conducted revealed that Flix 6 is considerably faster 173%(approx) to load a sequence of 100 panels, using artwork from a real production. Flix 6 enables a much better user experience with locating the appropriate sequences compared to Flix 5, also the underlying file transfer technology also delivers the assets to the end user faster and more reliably.

Asset Retrieval Time				
This test compares the time a Flix sequence is searchable, identifiable, and ready for use using				
Test Run	Flix Version	Number of Panels	Time Taken to Complete USER 1	Time Taken to Complete USER 2
1	5	100	16.82	25.00
2	5	100	31.29	17.00
3	5	100	13.98	14.00
4	5	100	7.86	14.00
5	5	100	7.10	16.00
			15.41	17.20
Test Run	Flix Version	Number of Panels	Time Taken to Complete	Time Taken to Complete
1	6	100	1.20	2.30
2	6	100	1.70	1.60
3	6	100	1.10	1.40
4	6	100	2.10	1.50
5	6	100	1.40	1.20
			1.50	1.60

Table shows the difference in search and retrieval times for 100 panels compared between Flix 5 and Flix 6

User Diaries Excerpts

"The Flix 6 UI is much snappier and less clunky than the old Flix 5 UI" - Stuart

5.2 Capacity and Scalability

This test is to assess the capacity and scalability of Flix, a vital component of an asset store.

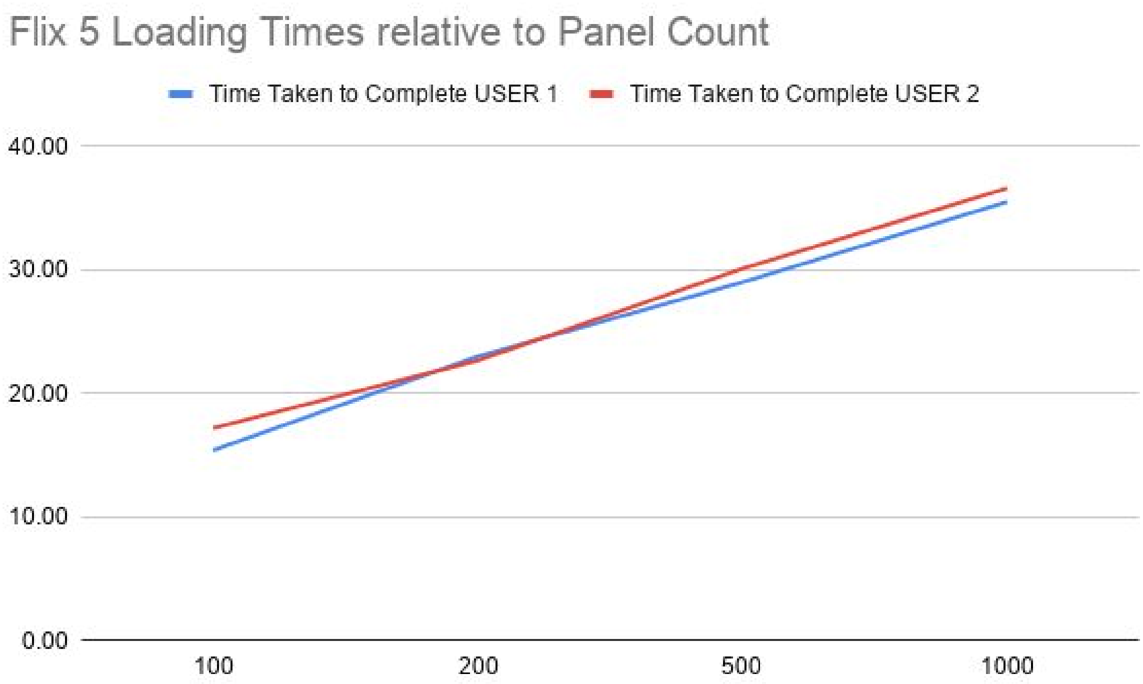
5.2.1 Test details

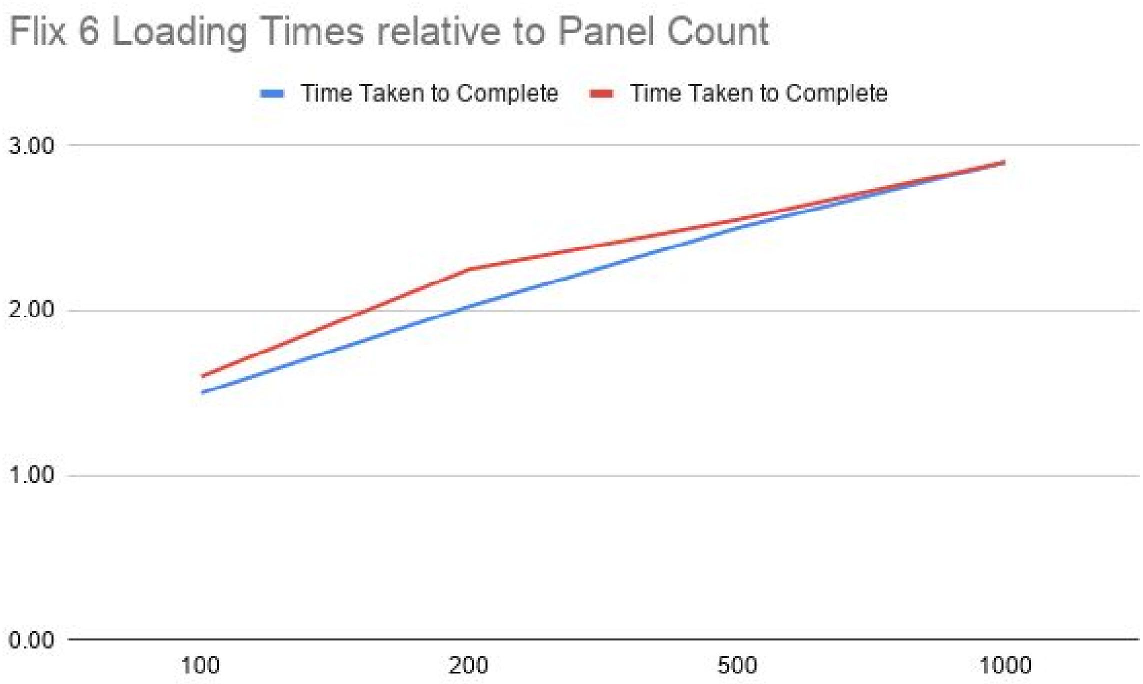
To carry out the test we asked 5 internal Flix users to follow a set of instructions. To remove discrepancies in build issues, all testing was carried out on Mac computers, with Linux Servers. The storage in these tests had the assets located on a network shared storage to emulate how Flix would be configured in a production environment.

Users were asked to load 3 different sequences which contained 10 panels, 100 panels, and 1000 panels and record the loading times and the overall user experience throughout the test. Participants were then interviewed after the testing to provide their feedback.

5.2.2 Results

The results of the capacity and scalability test concluded that Flix 6 did not show a discernible difference in loading times regardless of the amount of assets stored, or even the amount of assets within the current project being loaded. This is mostly down to the separation of assets and metadata, with metadata only increasing very slightly when there are larger datasets, and assets only being loaded on demand in Flix 6. Compared to Flix 5 in which the datasets including all existing assets are iterated over to gather the results, leading to a much greater linear increase in load times.





5.3 High Availability

The high availability testing is to ensure that the new redundancy features in Flix 6 provide enough resilience to withstand hardware outages within a studios datacenter. This ensures that Flix remains usable and there is no data loss during the outage. Ultimately saving a studio time and money.

5.3.1 Test details

Using an ablative approach to our test evaluation, we removed Flix servers from a live environment and monitored the users experience while servers were removed. This was designed to display Flix's new load balancing features which would ensure assets remain accessible even with partial hardware failures. This is a new feature of Flix 6, so evaluating and comparing results against Flix 5 would prove a total outage in all scenarios for Flix 5.

With Flix 6 running on 3 different physical servers, we asked the users to run through a normal usage workflow of creating artwork in Photoshop, and importing the new panels into Flix using the plugin tools. Then at an unknown time to the users we would remove a server from the cluster and monitor results. Users were asked to log their experience during the test.

Results and User Diaries Excerpts

The results were positive showing functionality and demonstrating redundancy of the product during outages and the termination of nodes in the node pool. Below are excerpts from users as evidence of this.

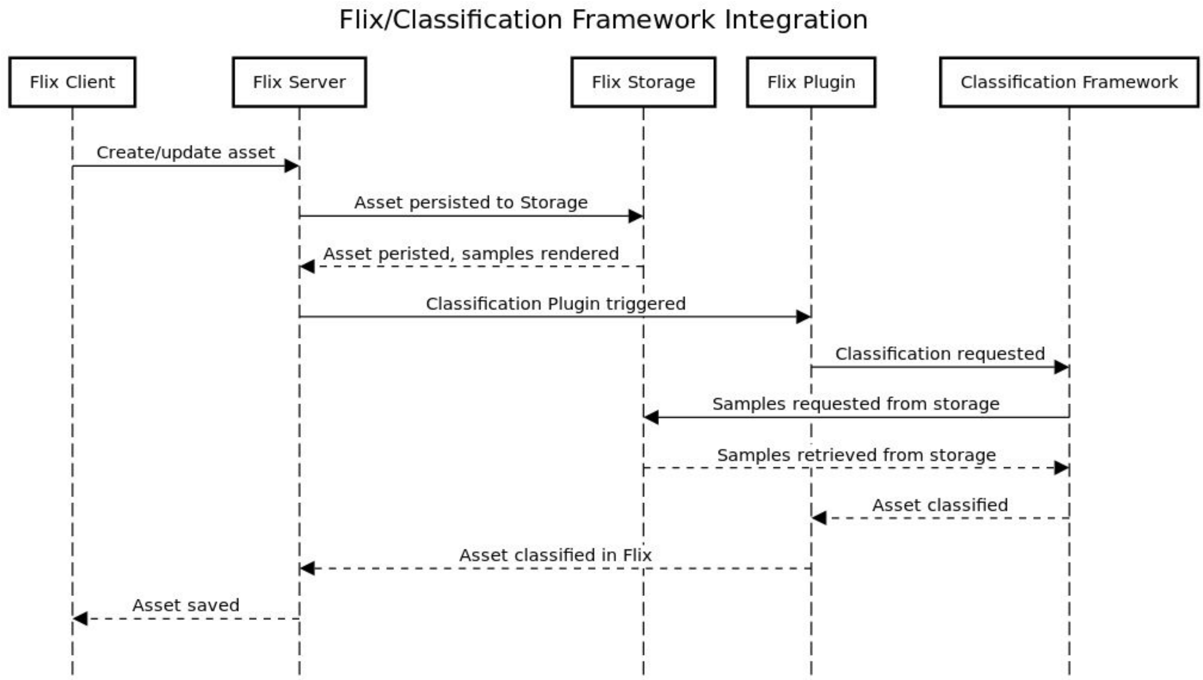
"I managed to import all panels during the test, and had no issues"
"Flix showed me a 'reconnecting' popup for a few seconds, while I was importing panels, but everything still worked"

5.4 Flix / Search & Transformation Interoperability Test

In D8.3 we described a couple of scenarios for testing the interoperability between Flix and the Search and Transformation Framework. These were amended slightly over the course of the evaluation in order to accommodate certain resources and constraints across the board. A number of components were developed that demonstrate the interoperability and integration between Flix and the Search and transformation framework.

5.4.1 Integrating Classification

In this scenario we integrated classification from the Search and Transformation Framework into an external instance of the flix asset management system, using its new plugin architecture, demonstrated and documented in D7.2. The diagram belows details the interaction between the various components as follows:



This can be summarised as follows:

1. The user uploads a new asset from the Flix Client to the Flix Server
2. The Flix Server persists the asset in the Flix Storage and renders any samples or thumbnails for classification
3. A classification plugin is triggered by the Flix Server when the asset storage is complete
4. The Classification Plugin makes a classification request to the Classification Framework
5. The Classification Framework requests samples from the Flix Storage and provides a classification response to the plugin
6. The Flix Server saves the labels in the classification response.

In order to deliver this interaction, a number of integration components were developed. These are detailed as follows:

Flix Classification Plugin

This was a plugin developed by DNEG that implemented the new plugin architecture as part of Flix 6.4. This is a python package which makes a classification request to the classification framework when a new asset is created or updated. It is responsible for transforming data about an asset in flix into the asset data model specified as part of the search and transformation framework, along with providing referencing to media objects in the flix storage, and then making a classification request using the base image HTTP API which is integral to every classifier and transformer. The package can be found here:

https://github.com/sauce-consortia/flix-publisher-plugin

The HTTP API used by the plugin is exactly the same as the API described in D9.8 specification report, and is uniform with all executions in the execution framework.

Flix Authentication and Media API

This was a Python library developed by DNEG that implemented the latest Flix HTTP API for authentication and media object access. This library authenticates against a Flix server using the HTTP API which is part of Flix6.4, with a set of client credentials, and creates an access token for subsequent requests. Additionally it will also re authenticate automatically when the token expires, providing a client with a seamless interaction with the Flix API's. Additionally, the library can also retrieve media objects from a Flix Server using the aforementioned token for authorization. The package for this can be found here:

https://github.com/sauce-consortia/lib-flix

This library has been integrated into a base classifier which classifies images and textures. It is further complemented by another flix plugin which creates image renders of Universal Scene Description. When a USD is uploaded to Flix, a USD rendering plugin generates image renderers of the USD. This library then gets samples of these renders from Flix using the Flix API and provides them to the image classifier for classification.

Results

Parts of the sequence and interaction detailed above were tested with the LED Wall dataset used for the Experimental Production in D8.4. We found that by integrating the classifier with this dataset, we could generate an increase of classification labels in excess of 100% for the entire dataset over the originally curated information. This small sample shows huge potential going forward, and a section in D9.8 Specification Report, covers this in more detail when describing the relationship with USD.

5.4.2 Integrating Flix Storage

This has been covered exclusively in D9.8 Specification Report in section 5. Flix Storage was used and tested as a platform primitive. The tests were entirely functional and objective, so were passed or failed as part of the libraries and projects associated with them. There was no subjective or usability testing done as part of this integration, however there was a degree of emphasis placed on further development options for the integration detailed in section 9.

6 Crowd and Path Animation Test

6.1 Crowd scene synthesis and metrics for quality evaluation

Deliverable 6.8 outlined a toolset and framework that leverages state-of-the-art techniques for the automated prototyping of re-usable crowd simulations. Section 4.6, “Metrics and Evaluation”, set out a plan for evaluating the toolset, broken into:

1. Laboratory testing of quantitative metrics (system stress testing and FPS/memory).
2. A plan for an expert review of the tools documented in this deliverable by external professionals.

We addressed the first point in D6.8 and we provide the implementation and results for the expert review of the tools by external professionals here.

6.2 Experiment Methodology

A heuristic problem discovery experiment design was implemented to qualitatively evaluate crowd simulation software documented in D6.8 using the Unity cross-platform game engine. This design approach applied Human-Computer Interaction methodologies specifically designed to explore the functionality, usability, and user experience of the tool. We also recorded the user sessions in order to calculate the time taken on individual tasks.

A remote participation procedure was used to record participant data. This approach to analysis was implemented due to the infeasibility of in-person participation due to COVID-19. Recruitment took place in the Republic of Ireland and UK for 2 months, from October to November 2020. Unity experts were targeted via fiverr.com, upwork.com, and advertisements posted to Irish game development groups and forums. Potential participants were also invited via direct email. A research information sheet was provided at this time, outlining the research motivations and execution procedures. Participants were encouraged to discuss the contents of the Research Information sheet in advance of the experiment. After the invited responses were processed, participants were allocated a date and time to remotely undertake the crowd simulation process.

6.2.1 Participants

The number of required participants was calculated as between 5 to 10 persons; providing an estimated problem discovery rate of 85.55% - 94.69% (Alroobaea & Mayhew, 2014). In this way, the experiment was able to increase context criticality and study complexity, while also ensuring and controlling for the design novelty of the proposed crowd simulation software (Macefield, 2009). A total of 6 participants were recruited to remotely take part in the experiment, consisting of 6 males and 0 females (n = 6). The average age of the group was 32 years old (SD = 7.4). All members of the pool were educated to the European Qualifications Framework (EQF) level 7 or above and were currently employed in the ISCO-08 employment categories of 251 (n = 3), 265 (n = 1), 214 (n = 1), and 133 (n = 1) with a total of 39 years experience (SD = 3.66) in their respective fields. To explore participant user types, each contributor was asked to describe their ability to use the Unity game engine (M = 4.4, SD = 0.80), identify their familiarity with creating crowd simulations in Unity (M = 2.20, SD = 1.47), and to describe their expertise in creating crowd simulations in Unity (M = 2.40, SD = 1.50) on fully labeled 5-point Likert scales. The self-identification questionnaire was issued to determine the skill level of the participants. The cohort identified themselves as technically competent, "Excellent" Unity users who were mainly novices (n = 5) at creating crowd simulations in the Unity game engine, with one advanced user (n = 1).

Participants 1, 3 and 5 were told to re-target from the first scenario to the second, whereas participants 2, 4 and 6 were told to start the second scenario from scratch.

6.2.2 Experiment Design

At the specified time and date of the experiment, participants were invited to remotely log into a project PC that was connected to the university network using TeamViewer (a software application for remote control, desktop sharing). For this purpose, a Dell Alienware Aurora R8 was set up as follows: Intel® Core™ i7-6700K CPU @4 GHz, 64GB RAM, 2 x GeForce RTX 2080 Super (Base clock: 1650 MHz, 8 Gb of GDDR6 Memory, and 3,072 CUDA cores), running Windows 10 Pro (1904), Visual Studio 2017 version 15.9.17, and Unity version 2018.4.12f1 (LTS). An average internet provider speed of Up = 901.41 Mbps (SD = 41.08), Down = 521.46 (SD = 11.35), and Ping = 1.6 ms (SD = 0.49) were measured at the PC before each session.

The study followed a two-step scenario testing strategy. The first scenario involved creating a simulated crowd scene and the second involved retargeting the crowd from the first scene to a semantically similar one. Two crowd simulation scenes were therefore created, each serving as a problem discovery measure for the new crowd simulation tools. This also allowed a comparison of re-targeting time between the two scenes. Users were provided with descriptions for what should be achieved in each scene, along with a brief video tutorial on how to use our tools in the Unity Editor. The tasks in the steps document were a decomposition of how to create the simulated crowd scene. Participants were allowed to ask questions about how to complete these tasks. Specific information on the result they should aim to achieve was also provided. The opportunity to discuss and analyze these procedures with a project researcher post-task ensured that the participants fully understood how the tools worked and could provide an informed evaluation. Participants were allowed a 30-minute break between the creation of each scene.

6.2.2.1 Experiment Scenario 1

Figure 1: Scenario 1 - Trinity College “Metropolis” Model

The first scenario uses the “Metropolis” 3D model [Figure 1]. The participants were asked to construct the scene with the following specifications:

1. Create 6 static groups of agents in the main square of the 3D model, containing 5, 4, 6, 3, 8, and 7 crowd members respectively. Each group should play an idle/chat/phone call/texting animation but not move locations throughout the simulation.
2. Create two crowd members with a walk animation that enter through the main front arch and navigate to a window in front of the “Rubric” game object. They should then trigger a “wash windows” animation.
3. Create 3 security guards that patrol the model. The security guards should practice social distancing.
4. Create a group of 4 people that make their way from the front arch to a grassy area and then perform yoga poses. These agents can then remain on the grass.
5. Create 2 groups of students with 10 in each group. Both groups should exhibit social distancing and should avoid static groups in the main square. The student groups should each play an “open door” animation when they reach their destination.
6. Create 18 members that wander randomly around the main square on navigable regions.
7. Create 3 crowd members that navigate to the water fountain and play a “drink” animation. They should then exit through the front arch.

6.2.2.2 Experiment Scenario 2

Figure 2: Scenario 2 - Love and Fifty Megatons Filmakademie model.

The second scenario uses the “Love and Fifty Megatons” 3D model [Figure 2]. The participants were asked to construct the scene with the following specifications:

1. Create 8 static groups of agents near benches along the main street, containing 5, 4, 6, 3, 4, 5, 3, and 4 crowd members respectively. Each group should play an idle/chat/phone call/texting animation but not move locations throughout the simulation.
2. Create two crowd members with suitable meshes that enter at the top of the street, make their way to locations around the “Cinema”, and play “sweep” animations.
3. Create a security guard that patrols on a predefined route. The security guard should practice social distance as much as possible.
4. Create a group of 4 people that make their way from one of the houses to a predefined location. They should then trigger a selection of dance animations.
5. Create 2 family groups with 5 in each family. Both groups should exhibit social distancing and should avoid static groups along the pavements. The families should move down the street to the cinema. A parent should trigger a “buying” animation when they reach the kiosk.
6. Create 12 crowd members that wander randomly along the pavements on either side of the road.
7. Create 3 crowd members that navigate to a picnic table and play an “eating” animation. They should then make their way back up the main street.

6.2.3 Recorded Metrics

On completion of the second scenario, participants filled out the Usability Metric for User Experience (UMUX-Lite), using 7-point scales for questionnaire continuity (Finstad, 2010; Lewis et al., 2015) and the User Experience Questionnaire (UEQ), measured using a 7-point Likert scale (Schrepp et al., 2017; Hinderks et al., 2019). The UMUX questionnaire was targeted toward the ISO 9241 definition of usability (effectiveness, efficiency, and satisfaction). The UEQ applied 26 separate question items, each with a seven-stage semantic differential scale. The UEQ scales measured the overall attractiveness of the crowd simulation tool as well as capturing user experiences across both classical usability (pragmatic qualities of efficiency, perspicuity, and dependability) and user experience (hedonistic qualities of originality and stimulation).

For each participant, the following data were also recorded during the scenario testing:

- Time on task for each of the steps 1-7, calculated from the screen recording data.
- Keystrokes and mouse clicks throughout the experiment.
- The results achieved by each participant, in the form of a Unity Scene file.

Periods of prolonged inactivity were removed from the recorded times by reprocessing mouse and keyboard activity. The final scene file, containing the newly developed crowd, was also saved to ensure consistency of quality between participants.

6.2.4 Results

The time on task for each participant is shown in table 2. Due to time constraints, some participants were not able to complete all tasks in a single remote session. These are marked with NA, where we asked participants to only complete a subset of the tasks for each scenario. Due to a technical error, the recording failed for participant 1 for tasks 5,6 and 7 but we were able to calculate the total time taken.

	Participant Number					
Task No.	**1**	**2**	**3**	**4**	**5**	**6**
1	0:33:34	0:24:20	0:45:51	0:37:06	0:02:01	1:26:50
2	0:46:26	0:34:20	0:39:16	0:58:03	0:26:50	1:35:00
3	1:11:55	0:55:31	0:50:40	0:33:33	0:04:14	0:40:40
4	0:39:02	0:18:42	0:47:18	0:40:00	0:29:25	0:50:20
5	0:32:43	0:27:30	0:37:24	0:59:00	0:08:23	NA
6	0:05:36	0:06:48	0:07:20	NA	0:10:00	NA
7	0:31:14	0:34:07	0:20:00	NA	0:07:01	NA
Total	4:20:30	3:21:18	4:07:49	3:47:42	1:27:54	4:32:50

Table 2: Time on task for experiment Scenario 1

	Participant Number					
Task Number	**1**	**2**	**3**	**4**	**5**	**6**
1	0:11:32	0:14:50	0:04:34	0:30:30	0:11:33	0:33:36
2	0:25:01	0:07:42	0:04:23	0:24:15	0:12:19	0:40:54
3	0:20:47	0:10:06	0:12:43	0:44:57	0:09:03	0:19:58
4	0:05:20	0:15:33	0:12:55	0:30:06	0:11:38	0:40:45
5	NA	0:31:08	0:37:25	0:19:05	0:24:42	NA
6	NA	0:07:00	0:02:14	NA	0:02:42	NA
7	NA	0:20:42	0:12:51	NA	0:11:10	NA
Total	1:02:40	1:47:01	1:27:05	2:28:53	1:23:07	2:15:13

Table 3: Time on task for experiment Scenario 2

Table 4 shows the means and standard deviations for time taken to create each scene, with granular data for the groups that could retarget components and those that could not retarget components.

	Scene 1 (Metropolis)	Scene 2 (LAFM)
Mean all participants	3:36:21	1:58:13
Mean can retarget	3:18:44	1:46:04
Mean can't retarget	3:53:57	2:10:22
Standard Deviation all participants	1:07:48	0:29:49
Standard Deviation can retarget	1:36:12	0:36:22
Standard Deviation can't retarget	0:36:10	0:21:21

Table 4: Means and standard deviations for participants to develop each scene, divided into those that could retarget components and those that could not.

Proportional time saving for each participant is shown in table 5, where green highlighting indicates that the participant was allowed to retarget components from the first scene in the second and red highlighting indicates that the participant was not allowed to retarget components from the first scene in the second:

Participant Number	1	3	5	2	4	6
% Time saving	43.19%	64.86%	5.44%	46.84%	34.61%	50.44%

Table 5: The proportional time saving achieved when creating scene 2, grouped according to whether the participant could retarget components from scene 1 or not.

The mean proportional time saving for participants who were allowed to retarget components from the first scene in the second was 37.83% and the mean proportional time saving for participants that were not allowed to use components from the first scene was 43.96%. However, these figures must be taken in relation to the standard deviations for the time saving achieved when creating the second scene, which are relatively high compared to the mean.

We anticipated that we would achieve a significant time saving only when allowing participants to retarget components to the second scene. However, the data suggests that this happens both when the participant can retarget and can not retarget. Upon reviewing the recordings of the participant sessions, we can suggest that the time saving achieved in both cases is a result of a number of factors, which we outline here:

1. A proportion of the time saving was due to effective retargeting of components as intended. This appeared to be especially true in tasks involving navigation, which offered large average proportional time savings (e.g. 61.8%, 54.17% and 23.7% for tasks 2,3 and 7 respectively)
2. For task 5, participants took longer to complete the task in the second scene. The video evidence suggested that this was random fluctuation (some participants decided to refine this task in the second scene more than the first).
3. Some participants accidentally mis-used our tools in the first scene which led to runtime errors. The person moderating the experiment had to step in to tell them what was going wrong. They didn't spend the same amount of time figuring things out in the second scene.
4. An increased understanding of the tools with use led to a decrease in the time taken to use them in the second scene.
5. The video evidence suggested that participants tended to cut corners more in the second scene than the first. We suggest that this was due to an increased desire to complete the second scene more quickly than the first in order to complete the overall experiment.

6. Most participants didn't classify themselves as crowd simulation experts with unity, which meant some of them learnt from the first scene, leading to a speed-up in the second scene.
7. Since this experiment was run remotely, there was occasionally a noticeable lag in the remote desktop software used for participants. The time taken for them to complete tasks varied correspondingly.

In order to get a true measure of time saving, some of these variables would need to be controlled for, which was beyond the capabilities of this study. We would suggest that future studies for a similar purpose would attempt to minimize the effect of these variables, or measure and account for them through a regression model.

The UMUX survey specifically targeted the usability of the crowd simulation tool by assessing its effectiveness, efficiency, and user satisfaction. An average UMUX-Lite Score of M = 68.06 (SD =11.20) was calculated using the formula: *(equ. 1)* $UMUX\text{-}Lite = 0.65*(UMUX(_{1,3}))+22.9$. Where $(UMUX(_{1,3}))$ refers to a UMUX score computed from Items 1 and 3 of the 7-point UMUX questionnaire (Lewis et al., 2013). As a benchmark for data validation purposes, a SUS comparison score of M = 67.14 (SD = 2.65) was also calculated using the formula: *(equ. 2)* $SUS\ comparison\ score = 0.65*((UMUX(_{1,3})-2)*(100/12))+22.9$. As a point of reference, Sauro and Lewis report that an average SUS score from 500 studies is a 68 (2016). Therefore, a SUS score above 68 would be considered above average and anything below 68 is below average.

The UEQ captured the users' immediate post-task impressions of the crowd simulation tool. The UEQ results for the attractiveness, pragmatic quality, and hedonistic quality of the crowd simulation tool can be seen in Table 6 and Figure 3. Each scale of the UEQ is ranged between -3 (extremely bad) and +3 (extremely good); however, in general, only values in a restricted range are likely to be observed due to central tendency bias. The attractiveness scale served as a valence dimension for the pragmatic and hedonic qualities. The overall "attractiveness" of the crowd simulation tool was considered to be 1.00 (SD = 1.06). The mean pragmatic qualities of the interaction were considered as 0.63, indicating that the practicality and functionality of the tool, when applied in this context, could achieve its intended goals. Furthermore, because of this score, it can be stated that the cohorts' satisfaction with the tool was also achieved; meaning that when using the software, they were able to sufficiently realize their personal goals. Moreover, this also indicated that the purpose of using the crowd simulation tool was clear and they understood how to use it effectively. The evaluation of the hedonic qualities scored high/low (M = 1.25); suggesting that the psychological and emotional experiences of the users were fulfilling. This score indicated that the participants were enthusiastic and that they enjoyed the overall experience. This also indicated that the memories and previous experiences of the users were positively evoked. The evocation of memories in this way signifies symbolic meaning from previous encounters within Unity and their personal experiences of crowd simulation during the experiment.

Scale	Mean	Standard Deviation	Confidence	Confidence interval	α
Attractiveness	1.00	1.06	0.85	0.15 - 1.85	0.93
Perspicuity	0.83	0.93	0.75	0.09 - 1.58	0.66
Efficiency	0.79	0.87	0.70	0.09 - 1.49	0.66
Dependability	0.25	1.04	0.83	0.58 - 1.08	0.82
Stimulation	1.71	0.80	0.64	1.07 - 2.35	0.83
Novelty	0.79	1.11	0.89	0.10 - 1.68	0.67

Table 6: Mean UEQ results with confidence intervals and Cronbach's Alpha-Coefficient per scale (n = 6).

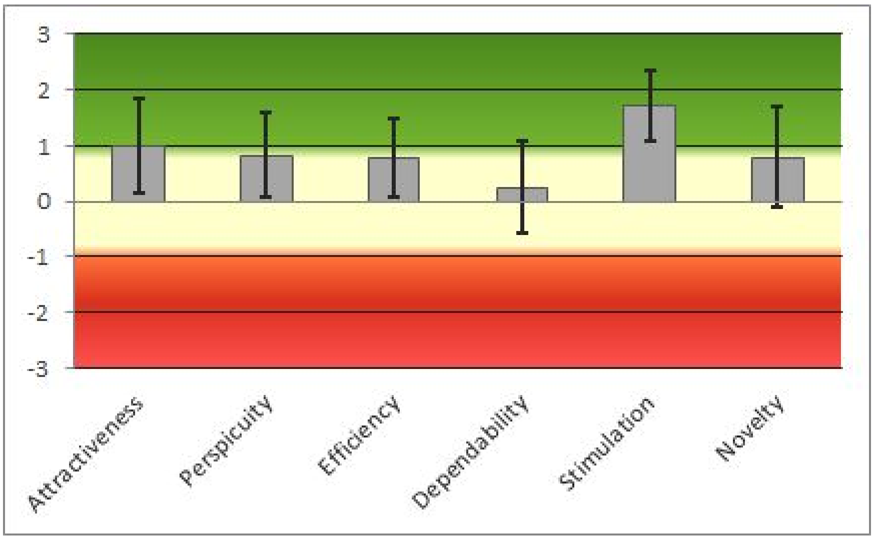


Figure 3: Mean values for user evaluations of the crowd simulation tool with error bars representing the 95% confidence interval.

6.2.5 Discussion of Results

From the presented findings, the following high-level conclusions can be drawn from our user-focussed problem discovery observations. When combined, both UMUX and UEQ post-task data reveal important data on how effective the simulation tool was and the overall users' experiences when completing crowd simulation tasks in Unity. The overall usability of the crowd simulation software was considered acceptable, with an average UMUX score of M = 68.06 (SD = 11.20) supporting this claim. Furthermore, the initial analyses of the UEQ indicated that the tool was attractive to use (M = 1.00; SD = 1.06), with positive evaluations for both hedonic (M = 0.63) and pragmatic (M = 1.25) qualities. Unpacking the lower hedonic qualities may further reveal the underlying reasons surrounding these evaluations and a discussion of these results is presented.

To begin with, the pragmatic aspects, or task-related quality of the tool, are detailed. The "efficiency" score ($M = 0.79$; $SD = 0.87$) reflected well on the cognitive resources demanded by the cohort when concerning the accuracy and completeness of the goals they had achieved during the different tasks. Furthermore, the users generally rated "perspicuity" highly ($M = 0.83$; $SD = 0.93$), indicating that they felt that the crowd simulation software was somewhat easy for them to become familiar with. This also suggested that it was easy for the users to learn how to use the software in the short amount of time they were given as well as the simulation methodology that was undertaken. In comparison, the UEQ measure of "dependability" was rated relatively low ($M = 0.25$; $SD = 1.04$), undermining the idea that the users felt that they were in control and suggesting that they felt insecure using the tool when undertaking the simulation tasks, as well as indicating that the system was perceptually unstable or behaved in an unpredictable way. This was true for all participants as the Alpha-Coefficient ($\alpha = 0.82$) suggested consistent measures. The hedonic, or non-task related qualities, user ratings of "stimulation" ($M = 1.71$; $SD = 0.80$) indicated that the extent to which the tool provided users with innovative and interesting functions, interactions, and stimulation was well received. The evaluation of "Novelty" ($M = 0.79$; $SD = 1.11$) also served to represent how innovative the cohort considered the crowd simulation tool to be. Therefore, due to the "newness" of the simulation tool, the novelty of the software was to be a compounding factor, one that may diminish over time. However, as our participants indicated that they were predominantly novices in crowd simulation activities, the novelty factor could be considered as a constant, yet influential, factor for the appraisal of crowd simulation software dependability moving forwards.

To unpack the participants' user experiences further, a follow up email was sent to the participants to retrospectively explore issues in relation to "dependability". Specifically:

- Which aspects of the crowd simulation toolkit did you feel were easy or difficult to control?
- Which components of the simulation toolkit did you find predictable or unpredictable?
- What modifications would you have liked to have made to the system to make it more dependable?

On the whole, participants felt that when using the crowd simulation tool it was easy to understand the purpose of each component and how they worked in conjunction with each other. However, they also felt that it was sometimes unclear which script they should use and for which purpose. Users felt that once the scene was set up, it was relatively easy to duplicate and modify for other groups of characters. Moreover, the cohort felt that the crowd simulation tool was fairly easy to add 'walk to' points and start an animation providing there were no obstacles. With obstacles, it was identified as being difficult to control the simulation as the moving characters chose to hug the predetermined line rather than avoid barriers and hurdles. In this area of concern, the users indicated that although the basic implementation of patrolling was easy to set up, they faced some issues with inconsistent prefabs. Users identified specific prefab issues with model scales being 2 or 3 times bigger than others and falling below the ground mesh. In this area of concern, some character prefabs responded as expected; however, others did not and in some cases walked beneath the ground mesh. These inconsistencies made the users feel that using the crowd simulation prefab assets were somewhat unpredictable in use.

For example:

> *"The use of the people prefabs was also a bit unclear - if I were doing a character from scratch, how would I add the appropriate scripts to the animator. (or maybe that was just the randomize script, was it simply triggering the state by name?)."*

> *"The use of goals was also a bit unintuitive: having to put in each goal twice as a start and end goal for each segment instead of just a simple list."*

> *"I didn't know how to stop the looped animations after they were triggered at the end goal."*

Other participants also highlighted having had issues with a prefab character's basic patrolling (semi-submerged), so it had to be switched with another character prefab that was previously used with no issues. Likewise, one participant had little success with the 'walk and sweep' function as the characters walked beneath the mesh and therefore could not trigger the sweeping animation. When this function was switched for 'walk and wash' they performed as expected; that is, the participant was required to duplicate 'walk and wash' and change the final animation for sweeping. This workaround was discovered mid-task, and the participant felt that there was no additional time to examine the assets or scripts in greater detail to establish where the issue was being generated.

Once the participants had gathered an understanding of how the crowd simulation tool functioned, they expressed that it was largely predictable and functioned as they expected. However, it was also identified that the user interface was unintuitive and complex. Furthermore, waypoint tracking was considered unpredictable on at least one occasion, and particularly unreliable when a character was moving quickly. When creating a patrol for the security guard one participant found that their characters would not take corners as expected. For example:

> *"I had plotted 4 points in a square shape on the corner of one building and expected the prefab to navigate from point to point, turning 90 degrees at each trigger. Instead, the character overshot the corners and looped around as if momentum was carrying it beyond the turn."*

Naturally, this brought the character off course. Additionally, when creating a patrolling character that should cross the road, it was not clear if the character would avoid the road mesh by default.

To make the crowd simulation tool more predictable, the participants identified specific areas of concern for improvement. For example, the toolkit as a whole needed to be more cohesive, as in having a central toolbar or window, as opposed to a collection of various components that appeared to be unintuitive concerning their requirements. To address this, it was suggested that the crowd simulation tool could add components automatically, as well as identifying the built-in Unity components that were also required for functionality. Furthermore, participants expressed that they would prefer to be able to set waypoints and have the character use a more standardized way to negotiate obstacles without following a strict line and defining the magnetic repulsion zones as being obstacles that should be negotiated rather than an absolute line to follow.

> *"The need of the other prefabs - the repulsors - was understandable, though the repulsor script itself was a bit unintuitive: i.e. whether to use multiple repulsor prefabs versus one prefab with multiple repulsor elements."*

By allowing for bespoke manipulation of the set waypoints, it would be possible to alter the navigation mesh dynamically. Other obvious suggestions were to standardize the prefab models, add visual indicators to establish when a prefab has triggered correctly (especially concerning the Y-axis), and clearer and more intuitive naming for assets and imported animations (route planner/route manager, static activity, etc.).

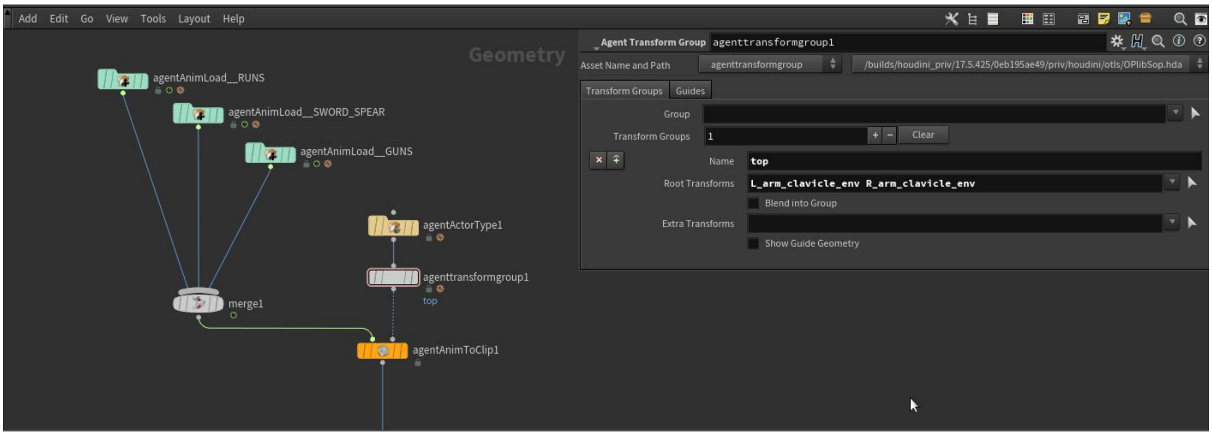
6.3 Path Animations

This section details the testing completed by DNEG as part of their Path Animations Evaluation Plan. It is broken into three main tests, a partial blending test, a granular placement test and a cheering crowd test. For each test, two artists were used to compare the time taken to complete the task without AIMS and then the time taken with AIMS. Additionally notes and feedback have been collected and collated from both artists and summarised below.

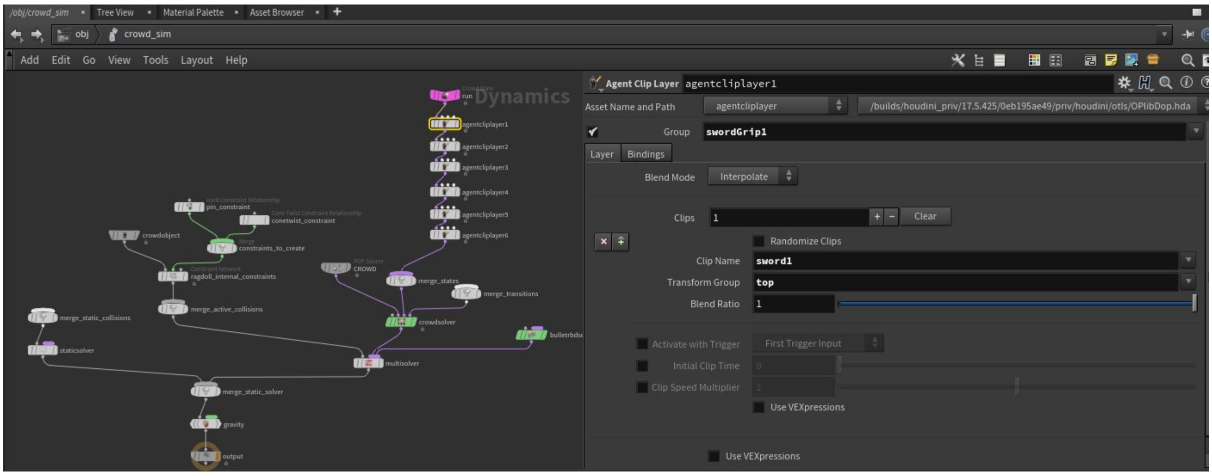
6.3.1 Partial Blending

This is a quite stereotypical situation found in 'crowd'. Sometimes it is because we want to add motion variation on top of an existing crowd pass while some other times we do not have clips for the exact actions that we need (and we end up combining two clips run + holding gun). In the suggested example we have a number of running clips that we are using on the agents independently from which battalion they belong to. However, the red individuals need to be holding fire arms whereas the green individuals are holding swords and shields. We are only going to focus on the partial blending so we will not invest time in doing complex steering etc in this example.
Time to prepare the test scene: 0.5 days
Time to prepare layout and basic setup: 1 hours

Solution 1 Feedback - without AIMS:
The main downside is the fact that the way Houdini offers to use motion layers, forces the artist to simulate the crowd (which leads to slow iteration times and less predictable results). For it to work, two main things need to be set up, the first one is the creation of a "Agent Transform Group" to isolate the arms joints. See image below:



The rest of the crowd setup is quite stereotypical, other than the fact that the artists have to create groups for the agents for each of the clips that needs to be added on top of the running motion. On this scene the artists are doing this via a point wrangle but any other method is valid as long as the artists create groups. In the DOP sim, one can define the clips to be added for each of the agent groups.



However, there is an additional price to pay due to the fact that the artists have to simulate. The artists can't use frame anchoring easily: what that means is that it is not easy to determine the exact position in space where the agents will be at any given frame. For this particular case the artists are just time-freezing the last frame of the simulation and compensating to make sure agents reach the desired location. This works for this case but it is not ideal since it assumes all agents in the simulation need to be at their goal location at the same global time.

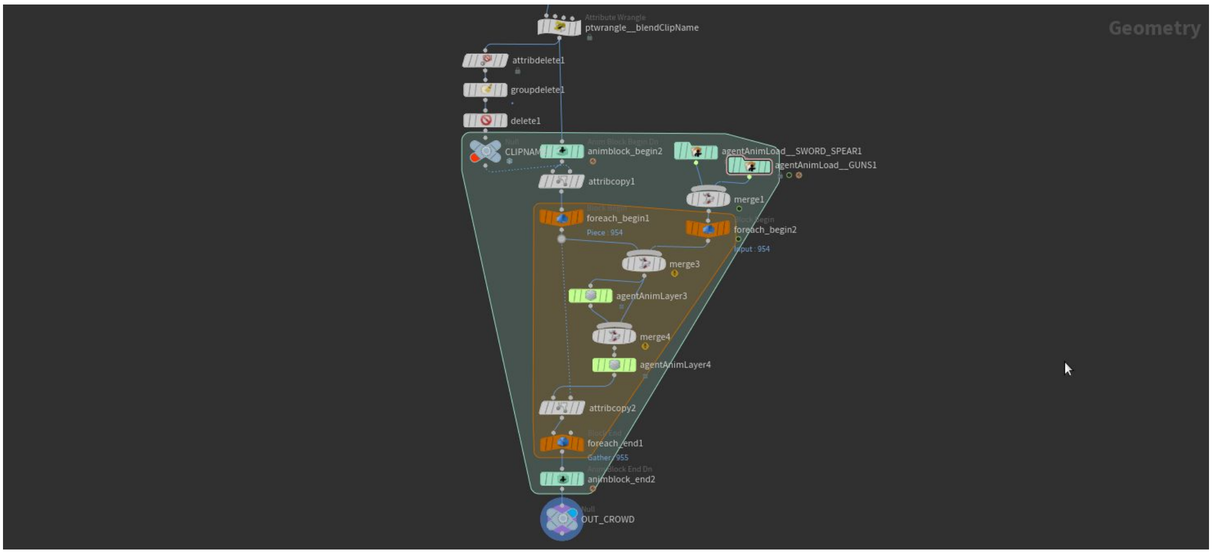
The biggest "time wasters" were having to simulate, having to compensate for the agents to reach their goal at a given frame.

Average Time to work / do a crowd pass on the shot: 3.5 hours

Solution 2 - with AIMS:

The biggest advantage of this approach is the fact that the locomotion resolution is independent from the partial blending (unlike the previous case where we had to simulate "just because" the artists wanted to make use of the AgentClipLayer DOP node). As a consequence, the work in the scene is effectively split in two stages. Once the artists are happy with the overall crowd sim, the artists can do passes on the additive blending without having to go back to simulating the locomotion.

To approach the shot this way, the main technology involved has been "Packed Anim Blocks" and "Agent Anim Layer". Due to the fact that these are core nodes, relatively low level, it has been necessary to do a bit of work to get the desired result (specific clip assignments, multiple blend joint chains, etc). The network where all the relevant logic is implemented can be seen below:



With this approach, most of the time went towards wiring up the network in the previous image. In this case, it took around the same amount of time to approach the shot with and without Aims. A higher level utility certainly helps and it is easy to see the advantages that it might have, compared to the previous approach (simulating). Also it's worth keeping in mind that the artist was not familiar with the technology, which also slowed things down to a certain degree.

An alternative version to this solution would be to modify the clips early and have them already on the agents' clip catalogs. However the artists tried to showcase the more usual use case where we end up having to do partial blending at a late stage (i.e. to address feedback).

Average Time to work / do a crowd pass on the shot: 3 hours

Results and Conclusions:

- Multiple blend chains are not possible on the node and multiple operations need to be chained.

- Getting an error on the anim block begin when using certain actor types.
- Forcing the upstream network to cook can be frustrating. An example had around 1K agents.
- Losing visual information within the block. Hard to tell which agent is which since only skeletons are visible and moved to the origin. Wrapper tools will help but question whether there is a way to at least not move the agents to the origin.
- Blended clips need to be provided fresh to the Agent Anim Layer node. It's not possible to blend two clips that are already on the agent.
- Hard to pass data into packed animations .
- Need to loop inside the packed animation block which prevents immediate visual feedback.

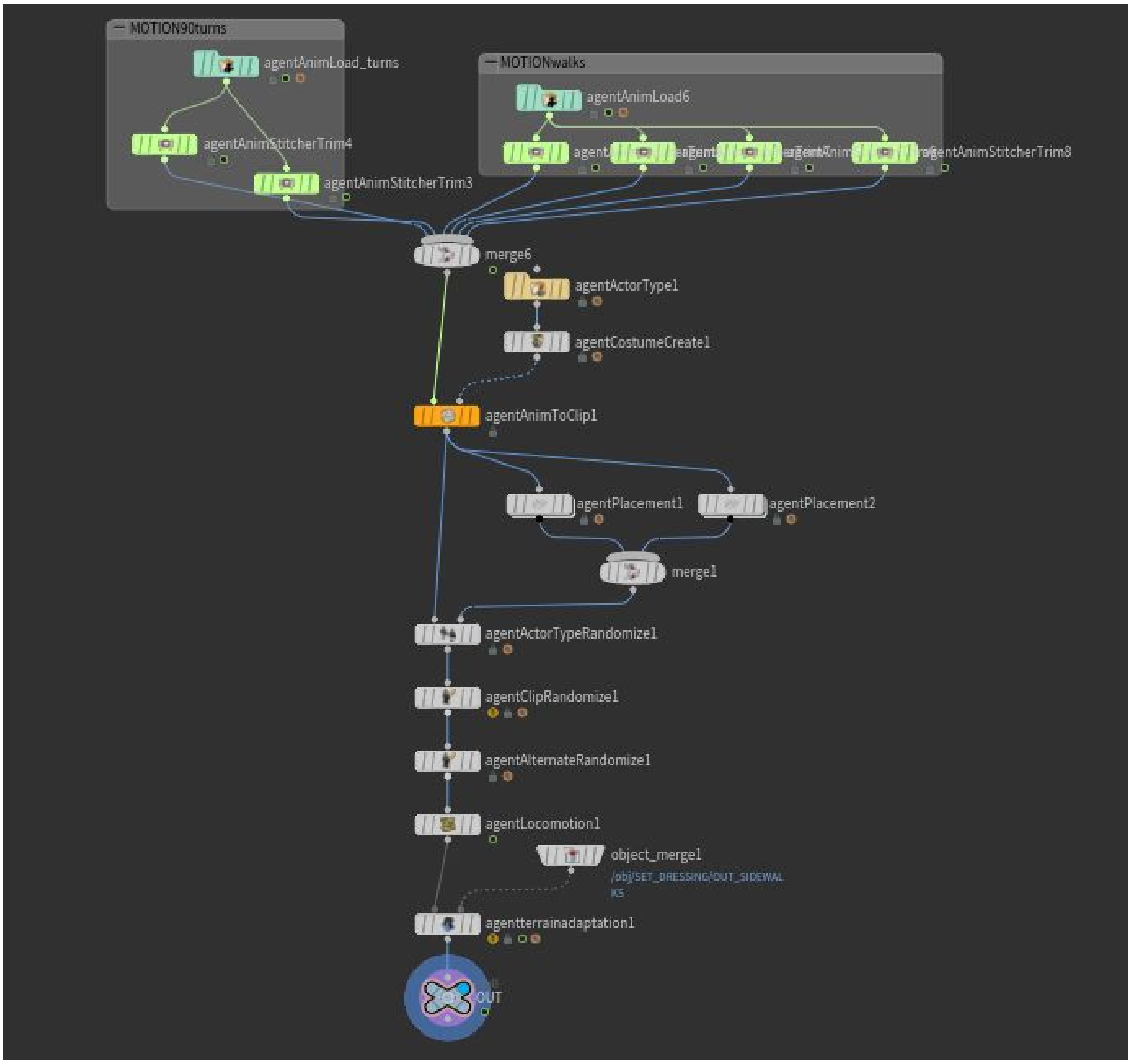
6.3.2 Granular Placement

This is also quite a common scenario we find in crowd. In the example use case the artists just have to populate the background with some generic motion of people walking, chatting, etc. In the suggested example we just have a handful of "walking straight" and "tight turn left/right" clips.

Time to prepare the test scene: 0.5 days
Time to prepare layout and basic setup: 1.5 hours

Solution 1 - without AIMS:

The workflow is pretty straightforward in this case. Below is the network used to achieve the result.



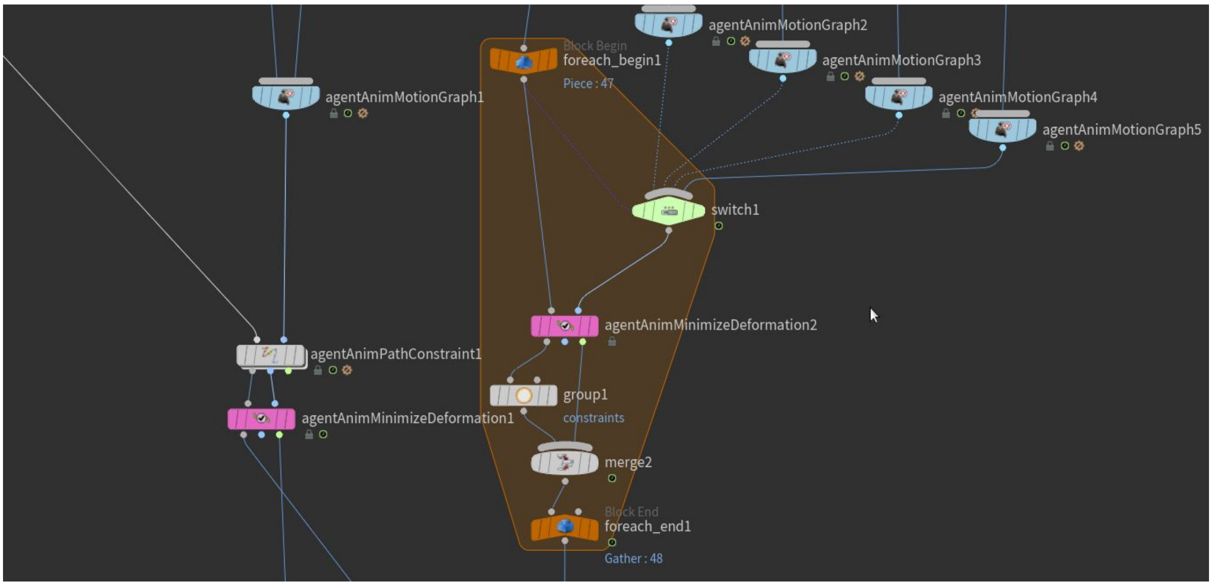
It is worth mentioning that the biggest time investment with this approach is having to pick clips and offsets individually for every placed source point. Although it is not too much of a problem in relatively small scenes this becomes tedious quickly if one needs to place and synchronise more complex crowds on larger shots. This is one area where AIMS could help since it could shield the user from the necessity of picking individual clips/time offsets manually

Average Time to work / do a crowd pass on the shot: 3.25 hours

Solution 2 with AIMS:

The biggest advantage of this approach is the fact that once motion is setup (more on this later) populating the scene is just a matter of drawing the paths where agents are necessary.

Due to current technical limitations it was hard to decouple the setup of the motion graph with the actual crowd layout work (see feedback section for more information). In this case the solution was to have a bunch of different motion graphs that could be picked accordingly for either straight walks or tight turns (the reason for the for loop on the image was to be able to have agents starting at different clip times since providing one motion graph currently causes the agents to start from the same moment on the clip).



Current limitations aside (ignoring the workarounds for achieving different starting clip frames, isolating animations, etc) and hoping performance holds for groups of ~50 agents this approach could be beneficial for use cases similar to the one tried here.

Average Time to work / do a crowd pass on the shot: 2.5 hours

Specific Notes and Feedback

- It is currently not possible to determine the starting frame of a placed anim (and/or having different starting animation frames getting picked up).
- Display flag vs node selection/template geometry behaviour. Would be great to make this default on nodes where what's interesting to see is the actual packed anims moving.
- It is really hard to isolate packed anims (ie. splitting or blasting). Prim/point numbers are not shown on viewport either.
- It is hard to work from camera or get a good sense of the motion quality with the skeleton view.
- Placement should consider terrain projection.
- Hard to get avoidance parameters to work as expected.
- Easy to introduce sliding (minimise deformation helps but there are still some situations where excessive sliding occurs).
- If one is using motion for male and female (different proportions) motion graphs need to be managed in isolation. If that needs to be the case perhaps it's a good idea for the placement tools to take different motion graphs.

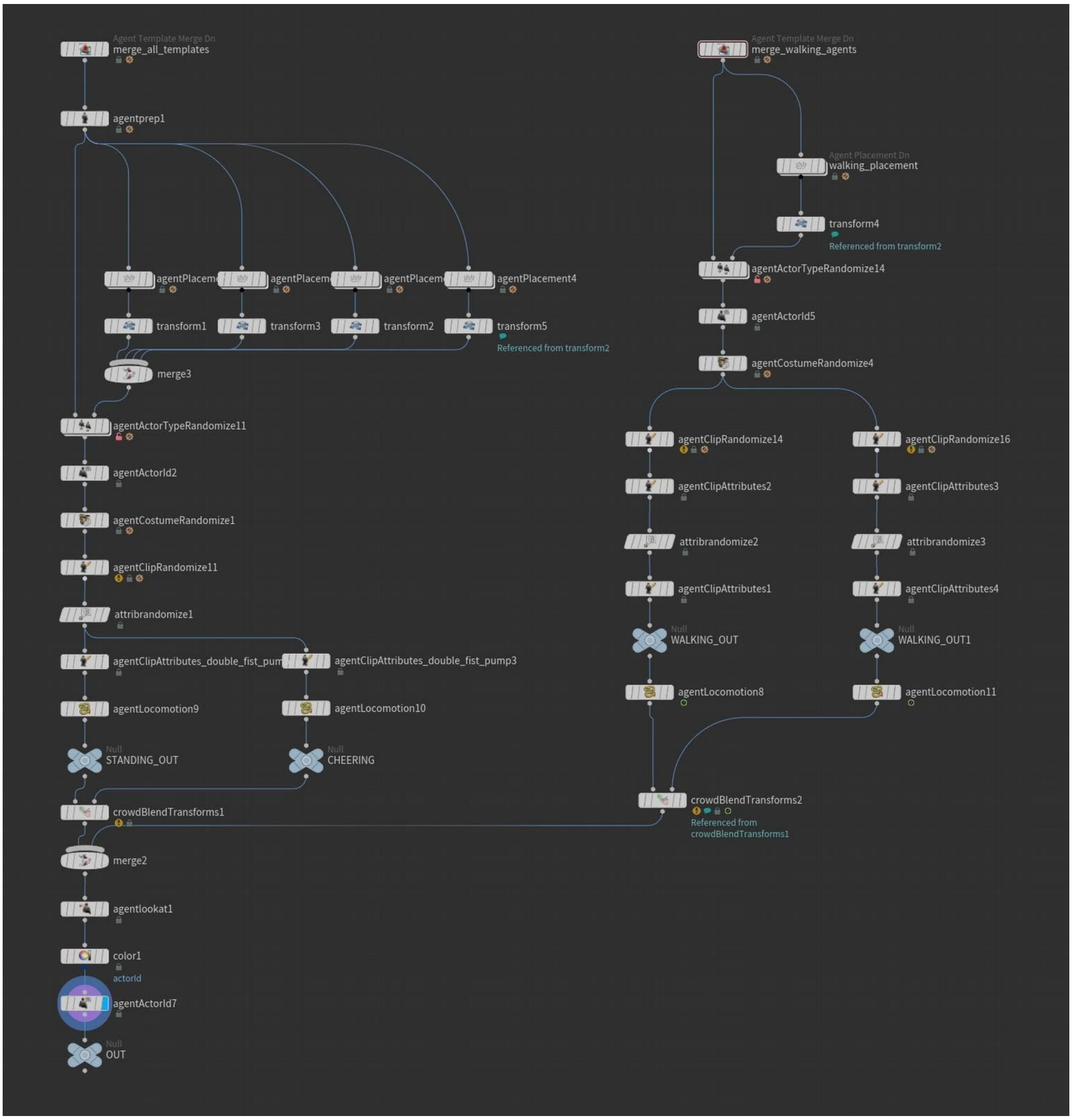
* For stability reasons the artists created constraints manually (this is to prevent crashes) but it also allowed identification of animations easier.

6.3.3 Cheering Crowd

This is also quite a common scenario we find in crowd. In the example use case the artists have a stationary crowd that transitions between multiple states. It's not good to just make one clip with slight offsets because that will generate a lot of twinning. Because of this the traditional setup has been to create one setup for each crowd state, and then blend the transforms between them. In this example, the artists don't just have standing crowd, but also a couple of walking characters. One difficulty with walking characters is that when blending the raw streams they can slide between their positions.

Time to prepare the test scene: 1 day
Time to prepare layout and basic setup: 4 days

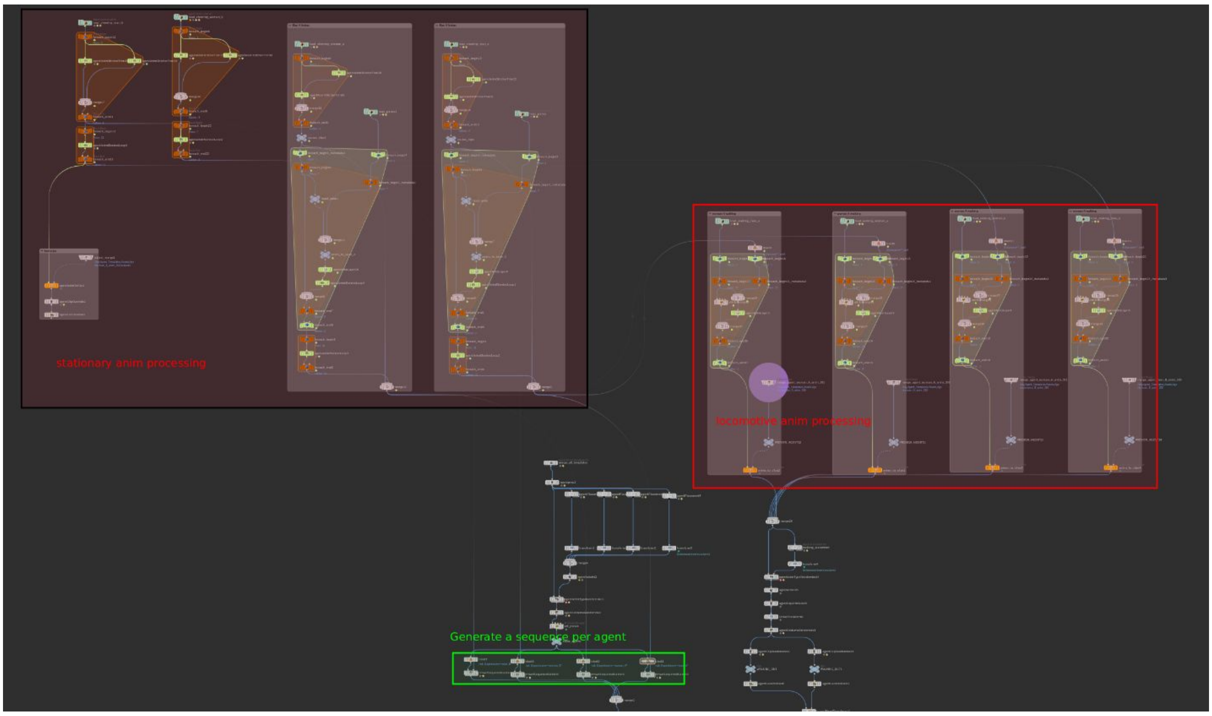
Solution 1 without AIMS



In this case for each locomotive speed, the artists have one agent stream with randomised clips. The artists then blend randomly based on a certain frame between the two streams. It gets a little more tricky when multiple states are used, and when layering animations.

Average Time to work / do a crowd pass on the shot: 8hrs

Solution 2 with AIMS:



In this case the artists tried to not use any dynamic blending at all. All animation is pre-generated with the AIMS sequence builder per agent. All randomization is done on an animation level. To achieve the variation needed for this the artists created many for-loops layering animation on top of the root animation. This would also have been possible with the previous example, so the red areas where the animations are generated would be needed there as well to get the same result.

The artists built a crowd sequence editor on top of the AIMS agent sequence editor. It loops over every agent primitive and randomizes the clip offsets, transitions lengths, state lengths, and clips using the same proximity variance as the clip randomize solution uses. The result was very similar to what the dynamic blend gave, and there was no noticeable difference in playback. We would imagine this solution would scale a lot better when faced with more than two states though, because then the artists wouldn't need a new crowd stream for every state. It would also work better for locomotive states transitioning to other locomotive states with different speeds where a blend would cause foot sliding.

Time to work / do a crowd pass on the shot: 8hrs

7 Conclusion

The evaluations and tests carried out in this deliverable clearly demonstrate both the individual and collective value of the work, research and outputs of the SAUCE project. A considerable amount of insight and progress was achieved through the course of the project and the consortium leveraged the capabilities of the partners to their fullest in order to deliver these results, even during challenging circumstances. Whilst many tests and evaluations provide huge insight and positive feedback, there is also evidence to suggest that further development and research on a number of initiatives could also be advantageous and valuable.

8 References

C. O'Sullivan and C. Ennis, "Metropolis: Multisensory Simulation of a Populated City," 2011 Third International Conference on Games and Virtual Worlds for Serious Applications, Athens, 2011, pp. 1-7, doi: 10.1109/VS-GAMES.2011.9

Alroobaea & Mayhew, 2014
https://www.researchgate.net/publication/266735808_How_Many_Participants_are_Really_Enough_for_Usability_Studies

Macefield, 2009
https://www.researchgate.net/publication/255600980_How_To_Specify_the_Participant_Group_Size_for_Usability_Studies_A_Practitioner%27s_Guide

Finstad, 2010;
https://www.researchgate.net/publication/220054775_The_Usability_Metric_for_User_Experience

Lewis et al., 2015
https://journals.sagepub.com/doi/abs/10.1177/1524839915580941

Schrepp et al., 2017
https://www.researchgate.net/publication/311982961_Construction_of_a_Benchmark_for_the_User_Experience_Questionnaire_UEQ

Hinderks et al., 2019
https://www.researchgate.net/publication/330735186_Developing_a_UX_KPI_based_on_the_User_Experience_Questionnaire

Houdini
https://www.sidefx.com/

Stemming and Lemmatization
https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html