978-89-294-1156-5

이미지 기반
원격 렌더링 제어 프로토콜 설계

2020. 11

김 민 아, 황 규 현, 이 세훈

한국과학기술정보연구원

목 차

1. 개요 ····· 1

2. 원격 렌더링 제어 프로토콜의 요구 사항 ····· 3
가. 요청 메시지 요구 사항 ····· 3
나. 응답 메시지 요구 사항 ····· 8

3. 프로토콜 시퀀스 ····· 9
가. Connect ····· 9
나. Health Check ····· 9
다. Load data ····· 10
라. Visualize ····· 11
마. Unload ····· 11
바. Close ····· 12
사. Error ····· 12

4. 프로토콜 메시지 개요 ····· 13
가. 메시지 헤더 ····· 13
나. 요청 메시지 종류 ····· 14
다. 응답 메시지 종류 ····· 16

5. 요청 메시지 상세 ····· 17
가. 이미지 ····· 17
나. 컨텍스트 ····· 18
다. 윈도우 ····· 19
라. 오브젝트 ····· 25

6. 응답 메시지 상세 ····· 32
가. 이미지 ····· 32
나. 상태 ····· 33

7. Predefined Types ····· 34
가. Primitive 타입 ····· 34
나. WKEY(Window Key) ····· 35
다. CAMERA ····· 35
라. STRING ····· 36
마. VKEY(Variable Key) ····· 36
바. LIST(Item) ····· 37
사. MAP(Key, Item) ····· 37
아. METADATA ····· 38
자. DCTX ····· 45

8. 결론 ····· 52

그림 목차

<figure 1> Remote control rendering on image based visualization system ·····2
<figure 2> Multiple visualization windows with two different data ·····4
<figure 3> Streamline particle animation example ·····5
<figure 4> Multiple slice visualization example ·····6
<figure 5> Connection sequence ·····9
<figure 6> Health check sequence ·····10
<figure 7> Load data sequence ·····10
<figure 8> Visualization sequence ·····11
<figure 9> Unload data sequence ·····12
<figure 10> Close sequence ·····12
<figure 11> Error sequence ·····12
<figure 12> General message ·····13
<figure 13> Message header octet string ·····13
<figure 14> RENDER request octet string ·····17
<figure 15> LOAD request octet string ·····18
<figure 16> LOAD STATE request octet string ·····18
<figure 17> COPY WINDOW request octet string ·····19
<figure 18> REMOVE WINDOW request octet string ·····19
<figure 19> SET BACKGROUND COLOR request octet string ·····20
<figure 20> SET BBOX VISIBLE request octet string ·····20
<figure 21> SET BBOX COLOR request octet string ·····21
<figure 22> SET COLORMAP request octet string ·····21
<figure 23> SET LEGEND VISIBLE request octet string ·····22
<figure 24> SET MIRROR request octet string ·····22
<figure 25> SET MIRRORED VISIBLE request octet string ·····23
<figure 26> START TIMESTEP ANIMATION request octet string ·····23
<figure 27> STOP TIMESTEP ANIMATION request octet string ·····24
<figure 28> NEW OBJECT request octet string ·····25
<figure 29> REMOVE OBJECT request octet string ·····25
<figure 30> SELECT OBJECT request octet string ·····26
<figure 31> SET OBJECT VISIBLE request octet string ·····27
<figure 32> SET OBJECT REPRESENTATION request octet string ·····27

<figure 33> SET CONTOURLINE THICKNESS request octet string ·····28
<figure 34> SET LINE REPRESENTATION request octet string ·····28
<figure 35> SET GLYPH SIZE request octet string ·····29
<figure 36> START PARTICLE ANIMATION request octet string ·····30
<figure 37> STOP PARTICLE ANIMATION request octet string ·····30
<figure 38> START SLICE ANIMATION request octet string ·····31
<figure 39> STOP SLICE ANIMATION request octet string ·····31
<figure 40> IMAGE response octet string ·····32
<figure 41> STATUS response octet string ·····33
<figure 42> WKEY octet string ·····35
<figure 43> CAMERA octet string ·····35
<figure 44> STRING octet string ·····36
<figure 45> VKEY octet string ·····36
<figure 46> LIST octet string ·····37
<figure 47> MAP octet string ·····38
<figure 48> AGGR octet string ·····38
<figure 49> BBOX octet string ·····39
<figure 50> MESH octet string ·····40
<figure 51> VAR octet string ·····40
<figure 52> BLOCK octet string ·····41
<figure 53> ELEM octet string ·····42
<figure 54> METADATA octet string ·····43
<figure 55> TR octet string ·····45
<figure 56> MIRROR octet string ·····46
<figure 57> COLOR octet string ·····46
<figure 58> COLORMAP octet string ·····47
<figure 59> OBJREP octet string ·····48
<figure 60> OBJ octet string ·····49
<figure 61> TKEY octet string ·····49
<figure 62> WINDOW octet string ·····50
<figure 63> DCTX octet string ·····51
<figure 64> GLOVE visualization system profiling result ·····52

표 목차

<table 1> Request messages for remote rendering ····· 6
<table 2> Response messages of remote rendering server ····· 8
<table 3> Message header specification ····· 14
<table 4> Remote rendering server request messages ····· 14
<table 5> Remote rendering server response messages ····· 16
<table 6> RENDER request specification ····· 17
<table 7> LOAD request specification ····· 18
<table 8> LOAD STATE request specification ····· 18
<table 9> COPY WINDOW request specification ····· 19
<table 10> REMOVE WINDOW request specification ····· 19
<table 11> SET BACKGROUND COLOR request specification ····· 20
<table 12> SET BBOX VISIBLE request specification ····· 20
<table 13> SET BBOX COLOR request specification ····· 21
<table 14> SET COLORMAP request specification ····· 21
<table 15> SET LEGEND VISIBLE request specification ····· 22
<table 16> SET MIRROR request specification ····· 22
<table 17> SET MIRRORED VISIBLE request specification ····· 23
<table 18> START TIMESTEP ANIMATION request specification ····· 23
<table 19> STOP TIMESTEP ANIMATION request specification ····· 24
<table 20> NEW OBJECT request specification ····· 25
<table 21> REMOVE OBJECT request specification ····· 26
<table 22> SELECT OBJECT request specification ····· 26
<table 23> SET OBJECT VISIBLE request specification ····· 27
<table 24> SET OBJECT REPRESENTATION request specification ····· 27
<table 25> SET CONTOURLINE THICKNESS request specification ····· 28
<table 26> SET LINE REPRESENTATION request specification ····· 29
<table 27> SET GLYPH SIZE request specification ····· 29
<table 28> START PARTICLE ANIMATION request specification ····· 30
<table 29> STOP PARTICLE ANIMATION request specification ····· 30
<table 30> START SLICE ANIMATION request specification ····· 31
<table 31> STOP SLICE ANIMATION request specification ····· 31
<table 32> IMAGE response specification ····· 32

<table 33> STATUS response specification ····· 33
<table 34> Primitive types ····· 34
<table 35> WKEY specification ····· 35
<table 36> CAMERA specification ····· 35
<table 37> VKEY specification ····· 36
<table 38> VKEY usage example ····· 37
<table 39> LIST specification ····· 37
<table 40> MAP specification ····· 38
<table 41> AGGR specification ····· 39
<table 42> BBOX specification ····· 39
<table 43> MESH specification ····· 40
<table 44> VAR specification ····· 41
<table 45> BLOCK specification ····· 42
<table 46> ELEM specification ····· 43
<table 47> METADATA specification ····· 44
<table 48> TR specification ····· 46
<table 49> MIRROR specification ····· 46
<table 50> COLOR specification ····· 47
<table 51> COLORMAP specification ····· 47
<table 52> OBJREP specification ····· 48
<table 53> OBJ specification ····· 49
<table 54> TKEY specification ····· 49
<table 55> WINDOW specification ····· 50
<table 56> DCTX specification ····· 51

1. 개요

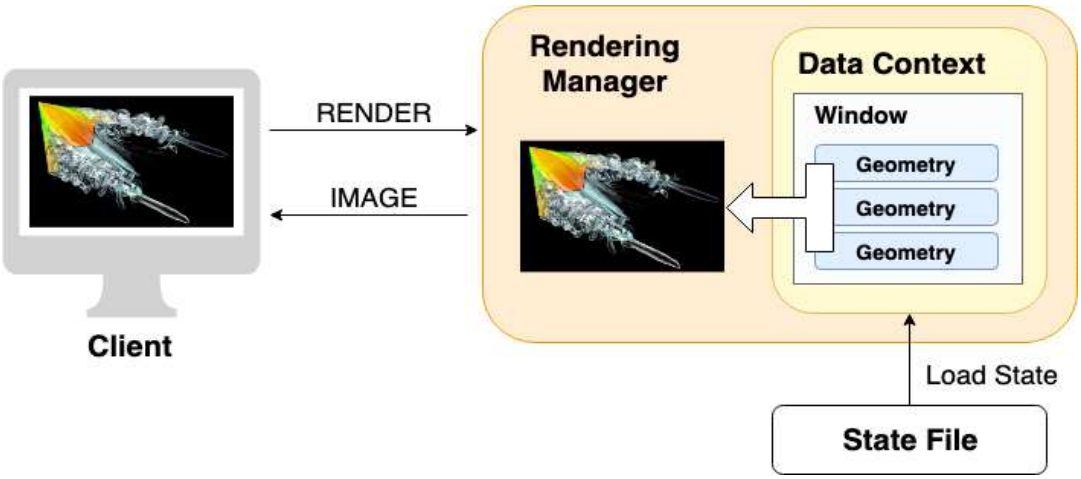
지구과학, 기상, 천체물리, 항공우주공학 등 과학과 엔지니어링 분야에서 전산 시뮬레이션은 복잡한 과학 현상을 모사하여 분석한다. 시뮬레이션 데이터의 분석 방법에는 그래프, 수치 데이터를 이용한 정량적 분석과 데이터의 가시화를 이용한 직관적이며 정성적인 분석의 방법이 존재한다. 우리는 가시화를 이용한 정성적 분석 방법으로 전체 데이터의 양상과 흐름을 알 수 있다. 따라서 전산 시뮬레이션이 올바른 방향으로 진행되고 있는 지에 대해 즉각적 판단이 가능하다.

컴퓨팅 자원의 급격한 성능 향상으로 전산 시뮬레이션은 좀 더 정밀하고 규모가 큰 실험이 가능하게 되었다. 이에 따라 시뮬레이션 결과 데이터의 양도 테라, 페타바이트 단위로 기하급수적으로 증가하고 있다. 시뮬레이션 데이터의 양이 증가함에 따라 과학적 가시화 관련 기술도 병렬 데이터 처리, 고해상도 렌더링, 복잡한 데이터를 알아보기 쉽게 보여주기 위한 표현 방법 등을 지원하도록 발전해 왔다. 또한 이들을 수행하기 위한 컴퓨팅 및 디스플레이 환경도 고성능, 고해상도를 필요로 한다.

그러나, 이러한 대용량 데이터의 가시화를 필요로 하는 실험실의 환경은 가시화를 위한 렌더링을 직접 수행할 수 있는 경우가 극히 드물다. 이 때문에, 가시화 도구들은 실험실에서 가시화를 요청하고, 렌더링은 고성능의 슈퍼컴퓨터가 있는 슈퍼컴퓨팅 센터에서 수행하여 그 결과를 다시 실험실로 전송할 수 있는 원격 렌더링을 지원한다. 그러나 원격 렌더링 요청은 가시화 도구에 의해 숨겨져 있으며, 가시화 도구에 특화되어 있다.

원격 가시화의 방법에는 두 가지가 있다. 폴리곤 기반의 가시화와 이미지 기반의 가시화이다. 폴리곤 기반의 가시화는 사용자가 원하는 데이터의 특징을 뽑아 그 결과를 메쉬(지오메트리)형태로 전송하는 것으로 폴리곤을 화면에 그리는 렌더링 과정은 클라이언트에서 수행된다. 반면 이미지 기반의 가시화는 사용자가 원하는 특징인 폴리곤을 뽑아내 렌더링까지 서버가 수행하며 그 결과

이미지만을 전송한다. 전송되는 데이터의 사이즌 폴리곤은 폴리곤 데이터의 크기이고 이미지는 이미지의 해상도이다. 폴리곤 데이터의 크기가 클수록 고정된 이미지 기반의 전송이 네트워크 부하를 줄일 수 있다. 이미지 기반 가시화에 있어 클라이언트와 서버가 상호 요청과 응답을 주고받으며 실시간으로 화면을 제어하는 방식은 <figure 1>과 같다. <figure 1>의 Geometry 는 가시화 결과 폴리곤의 지오메트리를 의미한다. 클라이언트가 요청을 하면 서버는 가시화 결과인 지오메트리를 렌더링 하여 이미지를 전송한다.



<figure 1> Remote control rendering on image based visualization system

본 연구에서는 한국과학기술정보연구원이 개발한 원격 가시화 도구 GLOVE의 원격 가시화 프로토콜 GIP에서 이미지 기반 가시화 결과를 전송하고 사용자의 조작에 따라 가시화 오브젝트에 변화가 생겨 다시 렌더링할 필요가 있을 경우 이를 원격에서 제어하기 위한 프로토콜을 설계하고 구현한다.

프로토콜의 구현을 위해 본 문서에서는 먼저 원격 제어 프로토콜의 요구사항을 분석하고, 요구사항을 만족시키기 위한 원격 렌더링 제어 프로토콜을 상세 설계한다. 마지막으로 이러한 프로토콜의 구현 및 실행 메커니즘을 설명한다.

2. 원격 렌더링 제어 프로토콜의 요구 사항

클라이언트의 화면에 가시화 결과가 표시된 후에 사용자는 화면에 그려진 오브젝트에 대해 여러 가지 동작(시점 변환, 오브젝트 표현 방식 변경 등)을 수행할 수 있다. 기존의 폴리곤 기반 가시화에서는 클라이언트가 이를 직접 수행했지만, 이미지 기반 가시화에서는 이에 대한 요청도 렌더링 서버로 전달하고, 새로운 이미지를 받아야 한다. 이를 위해 클라이언트와 서버 사이에는 사용자가 조작 가능한 모든 동작과 직접 대응되는 프로토콜을 설계하고 개발할 필요가 있다. 화면에 표시된 이미지에 변화가 일어날 수 있는 가시화 도구의 모든 사용자 조작을 정의하면 요청을 아래와 같이 분류할 수 있다.

가. 요청 메시지 요구 사항

□ **이미지**

이미지 기반 원격 제어 프로토콜의 가장 기본적인 요청으로 렌더링을 수행한 이미지를 요청한다.

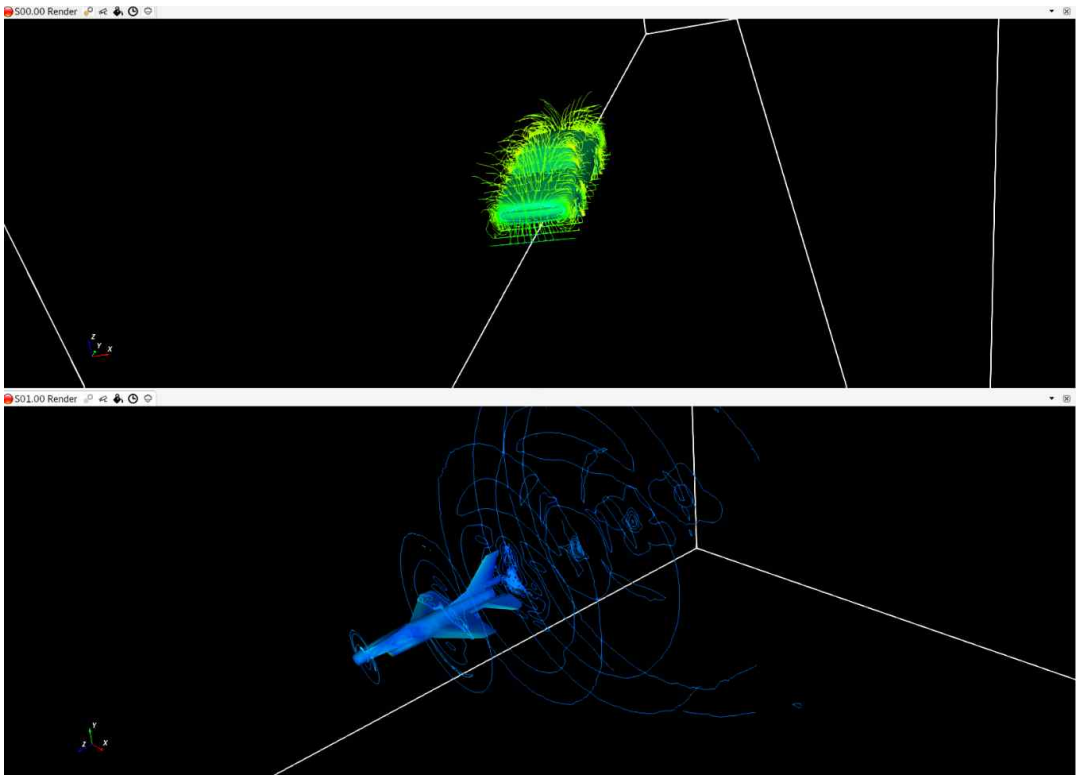
□ **윈도우**

사용자는 이미지를 그릴 윈도우를 만들 수도 삭제 할 수도 크기를 변경할 수도 있다. 또한 <figure 2>와 같이 현재 화면에 표현된 여러 개의 윈도우에 서로 다른 가시화 결과에 대한 이미지를 요청할 수 있다. 렌더링을 수행할 서버는 하나의 윈도우에 발생하는 변화 뿐 아니라 여러 개의 윈도우일 때 발생하는 알아야 하고 윈도우에 맞는 이미지를 전송할 수 있어야 한다.

□ **오브젝트**

사용자가 결과 이미지를 받은 후 가장 많이 조작하게 되는 것은 이미지 내의 결과 가시화 결과 오브젝트들이다. 새로운 가시화 요청에 의해 오브젝트가 새로 생기거나 존재하는 오브젝트를 선택하거나 숨기거나 지울 수 있다. 또한 만들어진 오브젝트의 속성을 변경할 수도 있다. 이러한 조작은 서버에게 알려져

야 하고 반영된 이미지를 새로 생성해야 한다.



<figure 2> Multiple visualization windows with two different data

□ 미러링 (Mirroring)

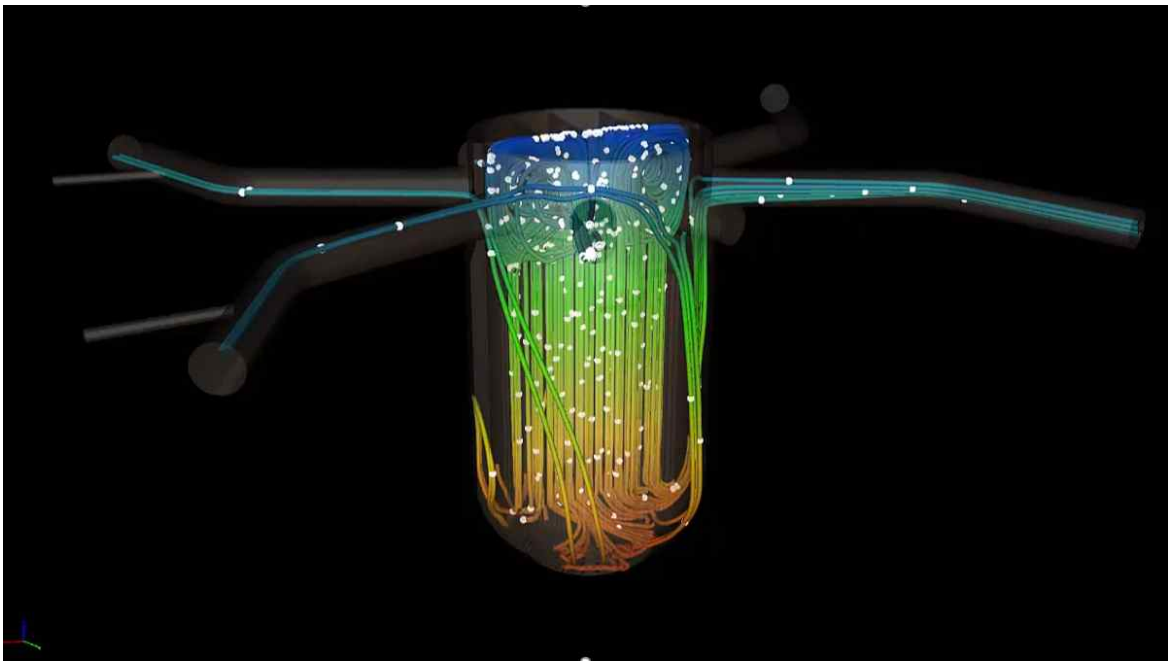
계산과학 시뮬레이션 중 일부는 전체 실험의 반만 수행할 때도 있다. 이를테면 항공기 시뮬레이션의 경우 오른쪽 날개를 시뮬레이션 한 후 동일한 결과를 미러링하여 왼쪽 결과로 함께 본다. 이럴 경우 하나의 가시화 결과를 축에 대해 미러링하는 특별한 가시화 요청이 발생하는데 서버는 이런 요청에 대한 이미지도 생성할 수 있어야 한다.

□ 타임스텝 애니메이션

시변환 시뮬레이션 데이터의 경우 시간의 변화에 따라 애니메이션 되는 가시화 결과를 보는 분석 방법이 수행된다. 이러한 경우 클라이언트는 다음 타임스텝에 대한 이미지를 요청함으로써 서버가 다음 타임 스텝의 이미지 렌더링을 수행할 수 있게 해야 한다.

□ 파티클 애니메이션

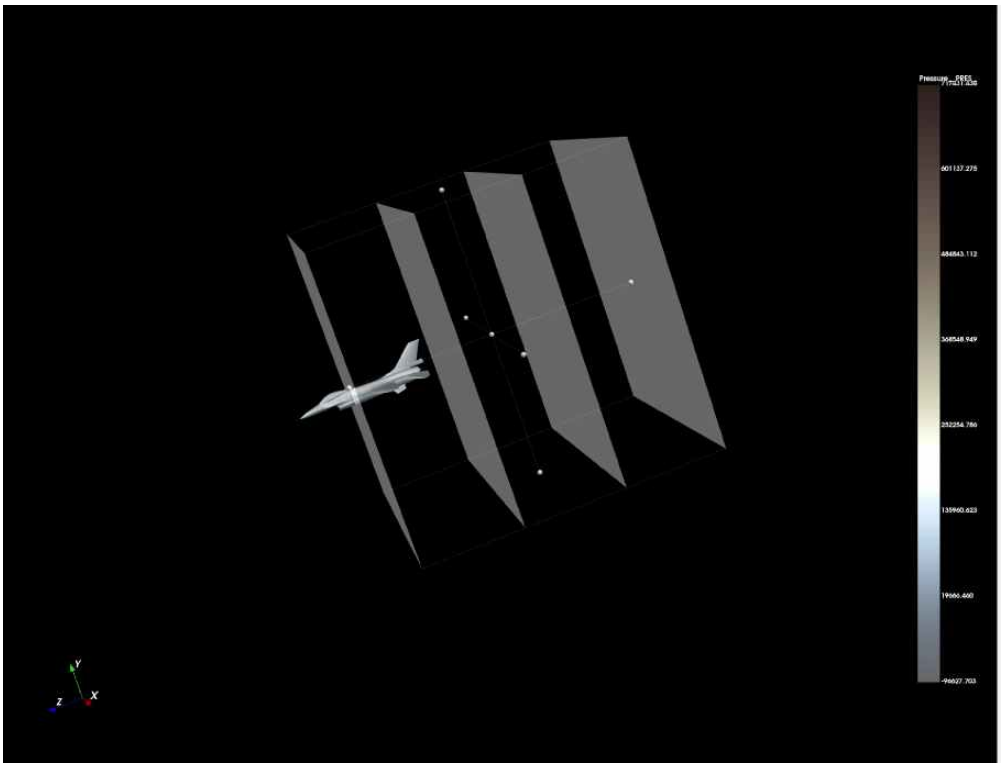
동일 타입 스텝 내에서의 애니메이션도 새로운 이미지 렌더링을 수행해야 한다. CFD 시뮬레이션의 경우 스트림라인 가시화 후에 만들어진 라인을 따라 작은 파티클을 흘려보내는 가시화 기법을 필요로 하는데 이 때, 파티클이 움직일 때마다 이미지 렌더링이 수행되어야 한다.



<figure 3> Streamline particle animation example

□ 슬라이스 애니메이션

가시화 기법 중 여러 개의 면으로 슬라이스를 한 후 슬라이스 한 면 위에 존재하는 데이터를 가시화하는 기법에 사용된다. 여러 개의 슬라이스 중 제일 앞의 슬라이스 가시화 결과부터 마지막 슬라이스까지 차례로 가시화 결과를 표시하는 애니메이션 기법으로 역시 새로운 이미지를 렌더링해야 한다. 이에 대한 요청을 서버에 전달하는 메시지도 필요하다.



<figure 4> Multiple slice visualization example

이상에서 정리한 요구사항에 의해 원격 렌더링 요청 메시지를 구분하여 정리하면 아래와 같다.

<table 1> Request messages for remote rendering

분류	기능	설명
이미지	RENDER	• 특정 윈도우의 이미지를 요청
컨텍스트	LOAD	• 데이터 셋을 최초로 로드
	LOAD STATE	• 상태(GST) 파일을 복원
윈도우	COPY WINDOW	• 특정 윈도우를 복사
	REMOVE WINDOW	• 특정 윈도우를 삭제
	SET BACKGROUND COLOR	• 배경화면 색상을 반영
	SET BBOX VISIBLE	• 데이터 전체의 바운딩박스의 화면 표시 여부를 수정
	SET BBOX COLOR	• 데이터 전체의 바운딩박스의 색상을 수정

	SET COLORMAP	• 특정 변수의 컬러맵 반영
	SET LEGEND VISIBLE	• 컬러 레전드의 화면 표시 여부를 반영
오브젝트	NEW OBJECT	• 오브젝트를 새로 생성
	REMOVE OBJECT	• 오브젝트를 삭제
	SELECT OBJECT	• 오브젝트를 선택
	SET OBJECT VISIBLE	• 오브젝트의 화면 표시 여부를 반영
	SET OBJECT REPRESENTATION	• 오브젝트의 표현 방식을 반영
	SET CONTOURLINE THICKNESS	• Contour 라인의 두께를 반영
	SET LINE REPRESENTATION	• 라인 오브젝트의 표현 방식을 반영
	SET LINE THICKNESS	• 라인 오브젝트의 두께를 반영
	SET GLYPH SIZE	• 글리프(Glyph) 오브젝트의 크기를 반영
미러링	MIRROR	• 미러링 적용
	SET MIRRORED VISIBLE	• 미러링 오브젝트의 화면 표시 여부를 수정
타임스텝 애니메이션	START TIMESTEP ANIMATION	• 타임스텝 애니메이션 시작
	END TIMESTEP ANIMATION	• 타임스텝 애니메이션 종료
파티클 애니메이션	START PARTICLE ANIMATION	• 라인 오브젝트의 파티클 애니메이션 시작
	END PARTICLE ANIMATION	• 라인 오브젝트의 파티클 애니메이션 종료
슬라이스 애니메이션	START SLICE ANIMATION	• 멀티 슬라이스 오브젝트의 슬라이스 애니메이션 시작
	END SLICE ANIMATION	• 멀티 슬라이스 오브젝트의 슬라이스 애니메이션 종료

나. 응답 메시지 요구 사항

클라이언트의 요청에 따라 서버는 해당하는 요청에 맞는 이미지를 렌더링하여 전송해야 한다. 그러나 서버의 응답은 해당 요청에 따른 이미지만 있는 것은 아니다. 클라이언트가 잘못된 요청을 할 경우, 혹은 서버가 이미지를 렌더링하는 동안 문제가 생겼을 경우 이를 클라이언트에 알려 줄 수 있어야 한다.

이를 위해 응답 메시지는 다음 두 가지로 구분할 수 있다.

<table 2> Response messages of remote rendering server

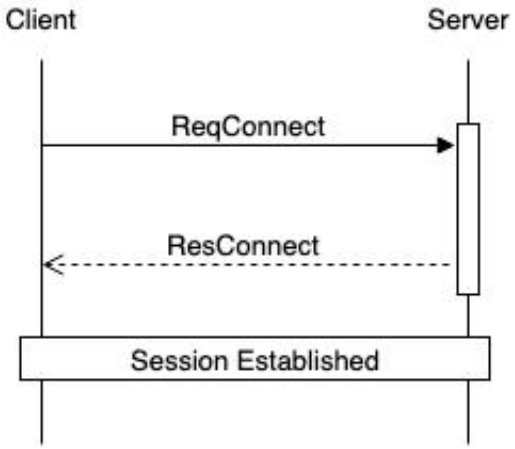
분류	기능	설명
이미지	IMAGE	• 특정 윈도우의 이미지를 전송
상태	STATUS	• 이미지를 제외한 일반 요청의 성공/실패 여부 전송

3. 프로토콜 시퀀스

원격 렌더링 제어 프로토콜은 KISTI 자체 개발 대용량 데이터 가시화 도구인 GLOVE의 가시화 프로토콜인 GIP의 일부로 전체 프로토콜 시퀀스는 GIP의 시퀀스에 이미지 요청 절차를 따른다. 이 때문에 원격 제어 프로토콜은 몇 가지 선행 프로토콜로 최초의 가시화 결과 이미지가 화면에 그려진 후 작동한다.

가. Connect

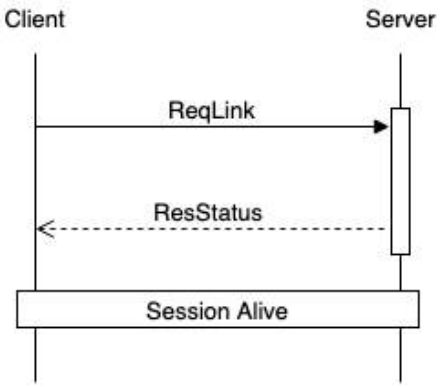
원격 렌더링 제어 프로토콜은 세션 유지 프로토콜이며, 이를 위해 클라이언트와 서버 간 커넥션 요청, 응답 과정 수행 후 원격 제어 프로토콜 메시지를 주고받을 수 있다.



<figure 5> Connection sequence

나. Health Check

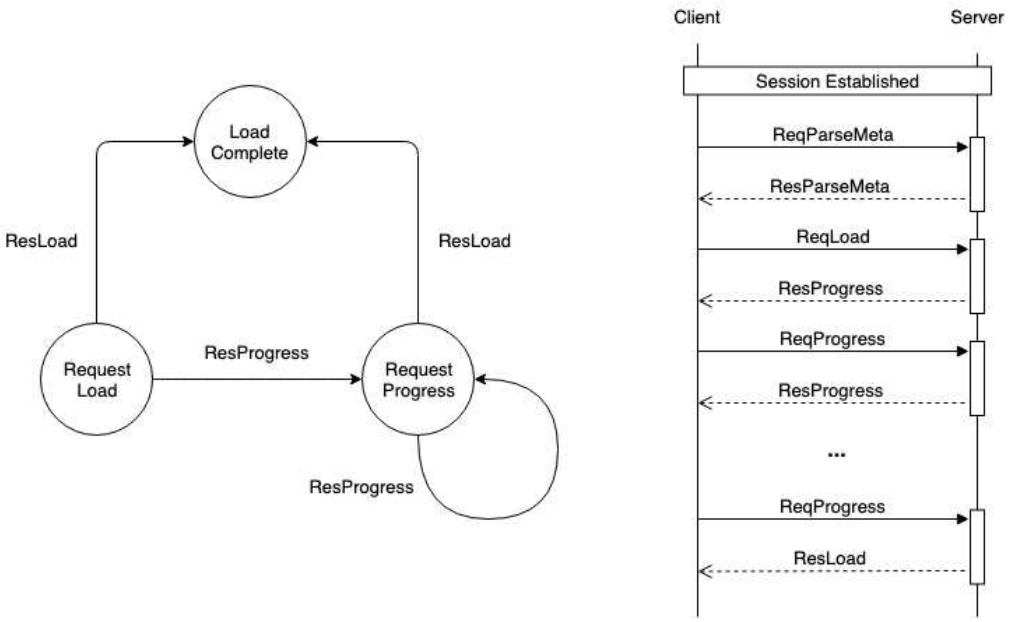
연결이 이루어지면 클라이언트는 서버와의 연결 상태를 확인하기 위해 주기적으로 연결 확인 요청을 보낸다. 연결이 비정상적으로 종료되면 클라이언트는 인터페이스에 적절하게 반영하여 사용자에게 이를 알리고, 연결을 정리한다.



<figure 6> Health check sequence

다. Load data

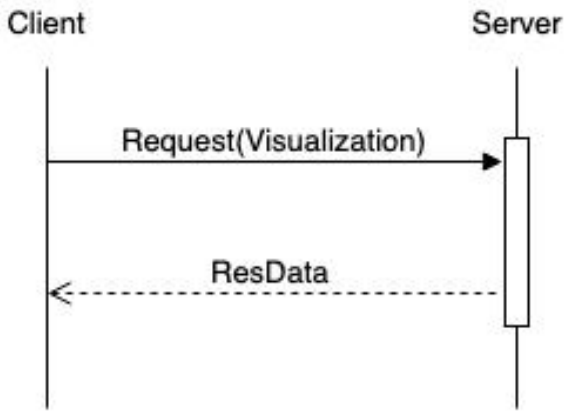
클라이언트는 가시화 요청 전에 데이터를 먼저 로드해야 한다. 데이터 로드는 세션 연결이 완료된 상태에서 요청할 수 있다. 데이터는 일반적으로 서버에 위치한다. 따라서 클라이언트는 데이터 셋의 위치를 명시하고, 요청하면 서버가 로드를 실행한 후 그 결과를 클라이언트에 전달한다. 대용량 데이터를 가시화하는 것이 필수적이므로 데이터 로드가 완료되기까지 클라이언트는 ReqParseMeta, ReqLoad, ReqProgress의 3단계를 거친다.



<figure 7> Load data sequence

라. Visualize

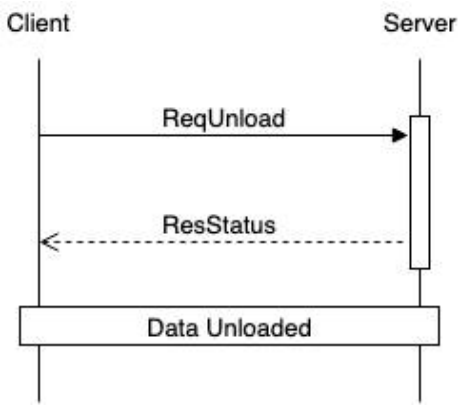
데이터 로드가 완료되면, 클라이언트는 가시화 요청을 보낼 수 있는 상태, 서버는 요청을 받을 수 있는 상태가 된다. 요청 메시지 내에 포함된 Primitive와 Type으로 가시화의 기능을 명시할 수 있으며, 일반적인 3차원 가시화 외에도 여러 가지 데이터에 대한 통계 정보 요청 및 그래프를 위한 데이터의 서브셋 요청 등이 가능하다. <figure 8>은 가시화 기능에 대한 기본적인 가시화 요청 처리의 한 예를 보여준다. Request(Visualization)으로 원격 렌더링 메시지 요청이 가능하다. 또한 결과 이미지는 ResData 4장에서는 이들 원격 렌더링 제어 메시지의 상세를 기술한다.

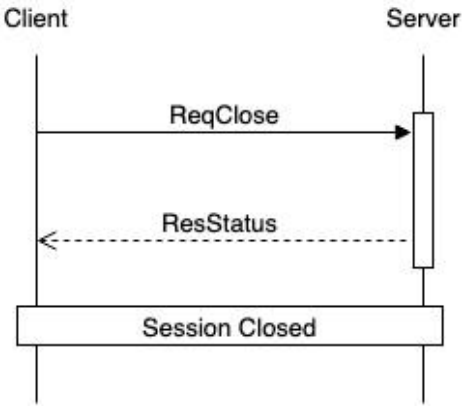


<figure 8> Visualization sequence

마. Unload

하나의 데이터 셋에 대한 가시화를 끝내고 다른 데이터 셋을 로드하기 위해서 클라이언트는 현재의 데이터 셋이 더 이상 필요 없다는 사실을 서버에게 알려줄 필요가 있다. 이때 클라이언트는 ReqUnload를 보낸다. 서버는 이 요청을 받으면 처리하고 있던 데이터 셋의 컨텍스트를 정리하고 새로운 데이터 셋을 읽어 들일 준비를 한 다음 ResStatus를 보내 새로운 데이터 셋을 로드할 준비가 되어 있다는 사실을 알린다. ResStatus는 성공일 때는 성공에 대한 코드를, 실패일 때는 상세한 에러 코드를 포함한다.





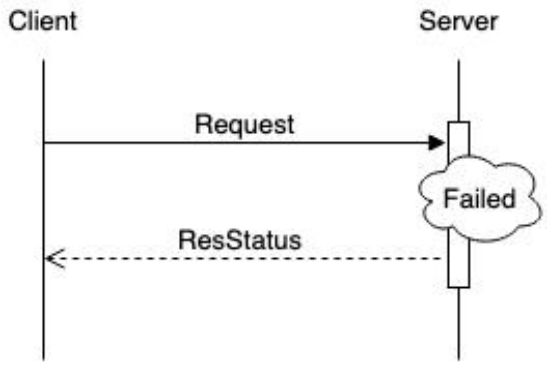
<figure 9> Unload data sequence <figure 10> Close sequence

바. Close

클라이언트가 연결 종료를 요청하면 서버는 연결을 해제하면서 저장해둔 세션 정보를 삭제한다. 다음은 세션이 종료되는 과정을 보여준다. 세션이 종료된 상태에서는 같은 TCP/IP connection으로 새로운 ReqConnect를 보내는 것과 TCP/IP connection을 종료하는 것이 가능하다.

사. Error

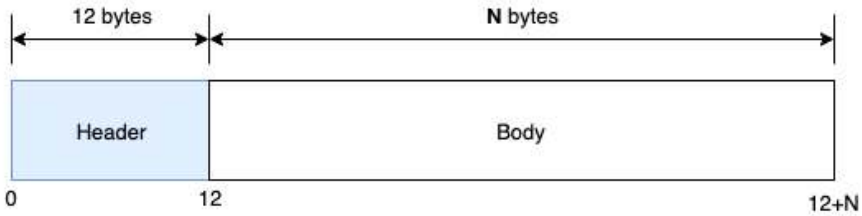
모든 요청은 서버에서 실패할 수 있다. 네트워크상의 문제가 아닌, 클라이언트와 서버가 정상적으로 연결된 상태에서 서버가 요청에 대한 응답을 만드는데 실패하면 서버는 ResStatus를 클라이언트로 전달한다.



<figure 11> Error sequence

4. 프로토콜 메시지 개요

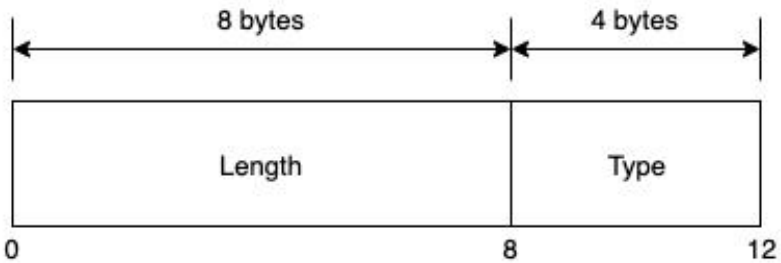
모든 메시지는 고정 길이의 헤더와 가변 길이의 바디로 이루어진다.



<figure 12> General message

가. 메시지 헤더

모든 메시지는 공통으로 메시지 헤더를 포함한다. 헤더는 고정 길이이며 바디의 형태와 길이에 대한 메타 정보를 포함한다.



<figure 13> Message header octet string

메시지의 종류는 크게 두 가지이다. 요청 메시지는 클라이언트가 렌더링 서버로 전달해서 서버의 액션이 일어나게 하는 메시지이고, 응답 메시지는 요청에 대한 서버의 답변이다. 클라이언트와 서버가 주고받는 모든 메시지에는 8바이트의 메시지 길이 정보와 8바이트의 고유 정보가 포함되어 있다. 메시지 길이는 길이 정보에 해당하는 8바이트를 제외한 나머지 데이터의 크기를 의미한다. 헤더에는 2개의 사전 정의된 필드가 포함되어 있다. 필드에 대한 목록과 설명은 다음과 같다.

<table 3> Message header specification

필드	타입	바이트	설명
Length	uint64_t	8	• Length를 제외한 나머지 데이터의 크기
Type	int32_t	4	• 메시지의 상세 종류를 구분

나. 요청 메시지 종류

2장에서 분석한 요구 사항에 따라 메시지의 종류에 코드를 부여하면 메시지의 종류는 다음과 같다.

<table 4> Remote rendering server request messages

분류	기능	코드	설명
이미지	RENDER	0x01	• 특정 윈도우의 이미지를 요청
컨텍스트	LOAD	0x02	• 데이터 셋을 최초로 로드
	LOAD STATE	0x03	• 상태(GST) 파일을 복원
윈도우	COPY WINDOW	0x04	• 특정 윈도우를 복사
	REMOVE WINDOW	0x05	• 특정 윈도우를 삭제
	SET BACKGROUND COLOR	0x06	• 배경화면 색상을 반영
	SET BBOX VISIBLE	0x07	• 데이터 전체의 바운딩박스의 화면 표시 여부를 수정
	SET BBOX COLOR	0x08	• 데이터 전체의 바운딩박스의 색상을 수정
	SET COLORMAP	0x09	• 특정 변수의 컬러맵 반영
	SET LEGEND VISIBLE	0x0A	• 컬러 레전드의 화면 표시 여부를 반영
	SET MIRROR	0x0B	• 미러링 적용
	SET MIRRORED VISIBLE	0x0C	• 미러링 오브젝트의 화면 표시 여부를 수정

	START TIMESTEP ANIMATION	0x0D	• 타임스텝 애니메이션 시작
	END TIMESTEP ANIMATION	0x0E	• 타임스텝 애니메이션 종료
오브젝트	NEW OBJECT	0x0F	• 오브젝트를 새로 생성
	REMOVE OBJECT	0x10	• 오브젝트를 삭제
	SELECT OBJECT	0x11	• 오브젝트를 선택
	SET OBJECT VISIBLE	0x12	• 오브젝트의 화면 표시 여부를 반영
	SET OBJECT REPRESENTATION	0x13	• 오브젝트의 표현 방식을 반영
	SET CONTOURLINE THICKNESS	0x14	• Contour 라인의 두께를 반영
	SET LINE REPRESENTATION	0x15	• 라인 오브젝트의 표현 방식을 반영
	SET LINE THICKNESS	0x16	• 라인 오브젝트의 두께를 반영
	SET GLYPH SIZE	0x17	• 글리프(Glyph) 오브젝트의 크기를 반영
	START PARTICLE ANIMATION	0x18	• 라인 오브젝트의 파티클 애니메이션 시작
	END PARTICLE ANIMATION	0x19	• 라인 오브젝트의 파티클 애니메이션 종료
	START SLICE ANIMATION	0x1A	• 멀티 슬라이스 오브젝트의 슬라이스 애니메이션 시작
	END SLICE ANIMATION	0x1B	• 멀티 슬라이스 오브젝트의 슬라이스 애니메이션 종료

다. 응답 메시지 종류

원격 렌더링을 위한 응답 메시지의 종류는 다음과 같다.

<table 5> Remote rendering server response messages

분류	기능	코드	설명
이미지	IMAGE	0xA1	• 특정 윈도우의 이미지를 전송
상태	STATUS	0xA2	• 이미지를 제외한 일반 요청의 성공/실패 여부 전송

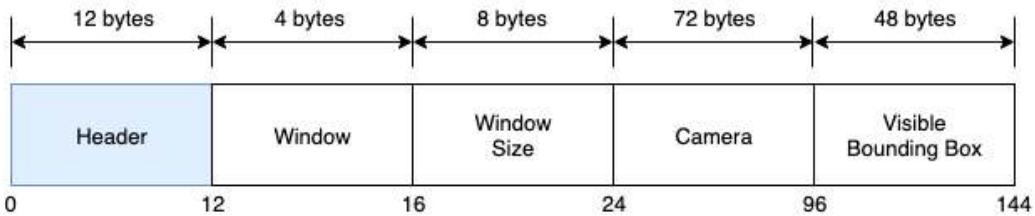
5. 요청 메시지 상세

본 여기에서는 요청 메시지의 상세 내용에 대해 알아본다. 요청 메시지는 크게 이미지, 컨텍스트, 윈도우, 오브젝트로 구분된다. 각 메시지의 명세는 필드, 타입, 바이트, 설명으로 구성된 표로 설명하며, 프로토콜에서 정의한 데이터 타입은 이탤릭체로 표기되며 상세 내용은 Predefined Types 절을 참조한다.

가. 이미지

□ RENDER

특정 윈도우의 현재 이미지를 요청할 때 사용한다. 클라이언트 윈도우의 Visible 바운딩박스 정보는 렌더링 서버 윈도우의 Visible 바운딩박스와 통합하여 카메라의 클리핑 범위를 새로 설정하기 위해 사용된다.



<figure 14> RENDER request octet string

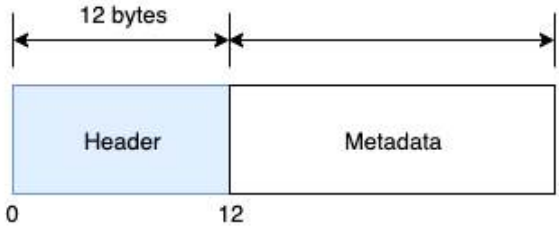
<table 6> RENDER request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 이미지를 요청할 윈도우
Window Size	uint16 [2]	8	• 윈도우의 크기(너비, 높이)
Camera	*CAMERA*	72	• 카메라 정보
Visible Bounding Box	double [6]	48	• 윈도우에서 현재 카메라에 보이고 있는 바운딩박스 정보(minX, maxX, minY, maxY, minZ, maxZ)

나. 컨텍스트

□ LOAD

데이터 셋을 최초로 로드하여 컨텍스트를 구성한다.



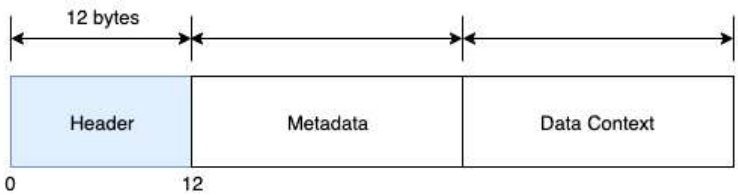
<figure 15> LOAD request octet string

<table 7> LOAD request specification

필드	타입	바이트	설명
Metadata	*METADATA*	-	• 데이터 셋의 메타데이터 정보

□ LOAD STATE

상태(GST) 파일을 복원하여 컨텍스트를 구성한다.



<figure 16> LOAD STATE request octet string

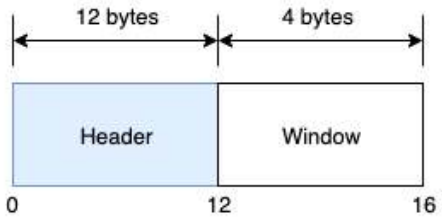
<table 8> LOAD STATE request specification

필드	타입	바이트	설명
Metadata	*METADATA*	-	• 데이터 셋의 메타데이터 정보
Data Context	*DCTX*	-	• 데이터 셋의 컨텍스트 정보

다. 윈도우

□ COPY WINDOW

특정 윈도우를 복제 요청한다.



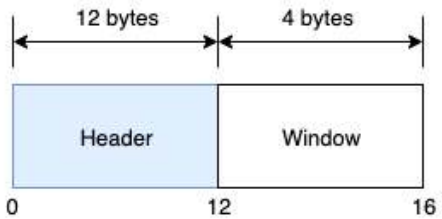
<figure 17> COPY WINDOW request octet string

<table 9> COPY WINDOW request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 복제를 요청할 윈도우

□ REMOVE WINDOW

특정 윈도우를 삭제 요청한다.



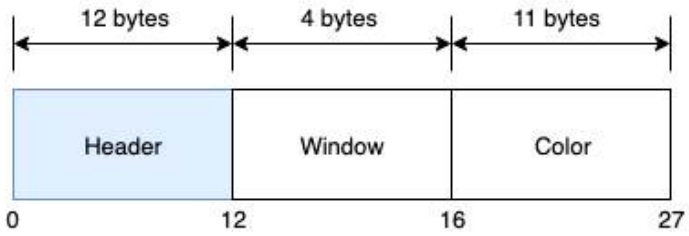
<figure 18> REMOVE WINDOW request octet string

<table 10> REMOVE WINDOW request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 삭제를 요청할 윈도우

□ SET BACKGROUND COLOR

특정 윈도우의 배경화면의 색상을 변경한다.



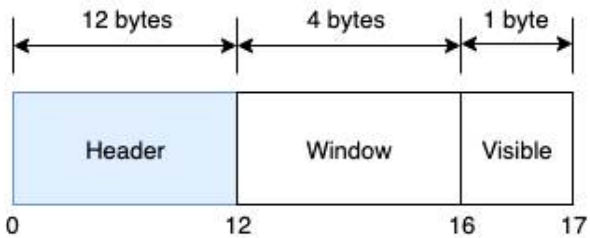
<figure 19> SET BACKGROUND COLOR request octet string

<table 11> SET BACKGROUND COLOR request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 배경화면의 색상을 변경할 윈도우
Color	*COLOR*	11	• 변경할 색상

□ SET BBOX VISIBLE

특정 윈도우의 바운딩박스의 표시 여부를 변경한다.



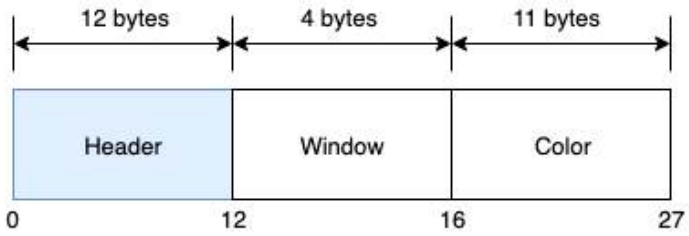
<figure 20> SET BBOX VISIBLE request octet string

<table 12> SET BBOX VISIBLE request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 바운딩박스의 표시 여부를 변경할 윈도우
Visible	bool	1	• 표시 여부

□ SET BBOX COLOR

특정 윈도우의 바운딩박스의 색상을 변경한다.



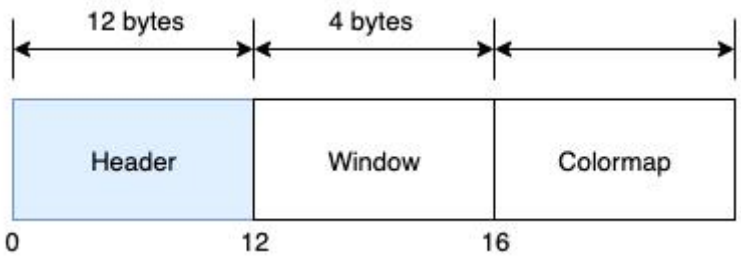
<figure 21> SET BBOX COLOR request octet string

<table 13> SET BBOX COLOR request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 바운딩박스의 색상을 변경할 윈도우
Color	*COLOR*	11	• 변경할 색상

□ SET COLORMAP

특정 윈도우의 특정 변수에 대한 컬러맵을 변경한다.



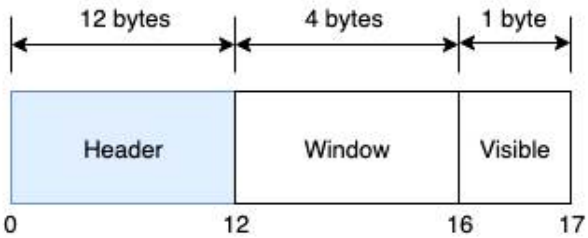
<figure 22> SET COLORMAP request octet string

<table 14> SET COLORMAP request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 컬러맵을 적용할 윈도우
Colormap	*COLORMAP*	-	• 컬러맵 정보

□ SET LEGEND VISIBLE

특정 윈도우의 컬러 레전드의 표시 여부를 변경한다.



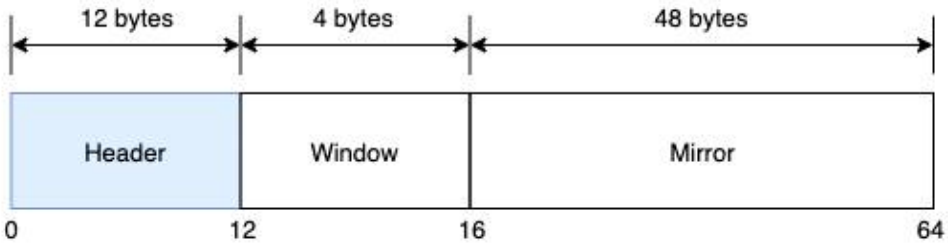
<figure 23> SET LEGEND VISIBLE request octet string

<table 15> SET LEGEND VISIBLE request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 컬러 레전드의 표시 여부를 변경할 윈도우
Visible	bool	1	• 표시 여부

□ SET MIRROR

특정 윈도우에 미러링을 적용한다.



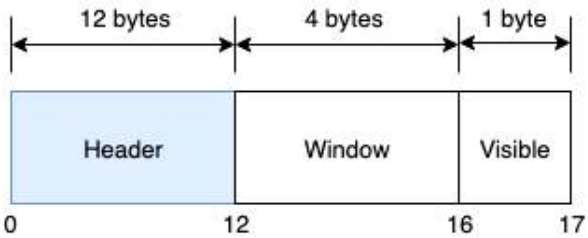
<figure 24> SET MIRROR request octet string

<table 16> SET MIRROR request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 미러링을 적용할 윈도우
Mirror	*MIRROR*	48	• 미러링 평면 정보

□ SET MIRRORED VISIBLE

특정 윈도우의 미러링 오브젝트의 표시 여부를 변경한다.



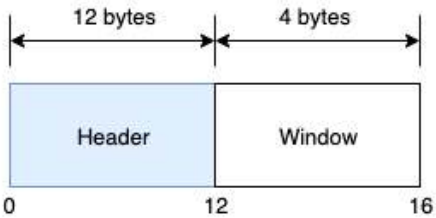
<figure 25> SET MIRRORED VISIBLE request octet string

<table 17> SET MIRRORED VISIBLE request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 미러링 오브젝트의 표시 여부를 변경할 윈도우
Visible	bool	1	• 표시 여부

□ START TIMESTEP ANIMATION

특정 윈도우의 타임스텝 애니메이션을 시작한다.



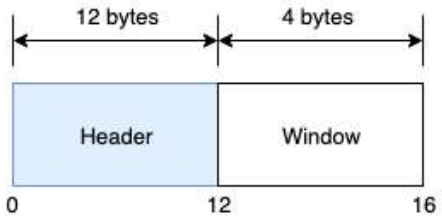
<figure 26> START TIMESTEP ANIMATION
request octet string

<table 18> START TIMESTEP ANIMATION request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 타임스텝 애니메이션을 시작할 윈도우

□ STOP TIMESTEP ANIMATION

특정 윈도우의 타임스텝 애니메이션을 종료한다.



<figure 27> STOP TIMESTEP ANIMATION
request octet string

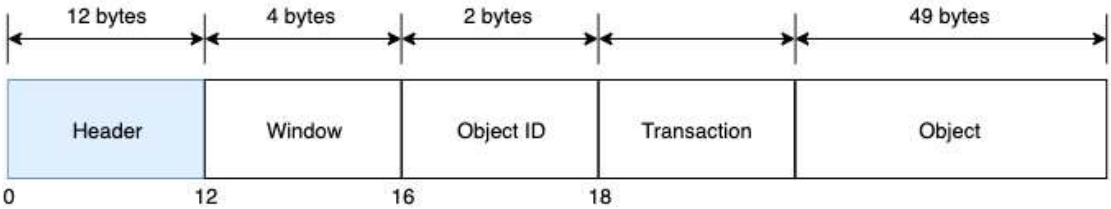
<table 19> STOP TIMESTEP ANIMATION request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 타임스텝 애니메이션을 종료할 윈도우

라. 오브젝트

□ NEW OBJECT

오브젝트를 새로 생성한다.



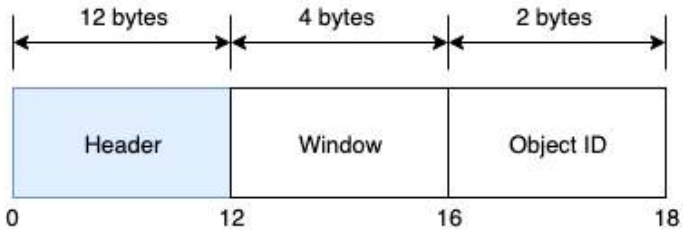
<figure 28> NEW OBJECT request octet string

<table 20> NEW OBJECT request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 오브젝트가 속한 윈도우
Object ID	*OID*	2	• 오브젝트 아이디
Transaction	*TR*	-	• 오브젝트의 트랜잭션 정보
Object	*OBJ*	49	• 오브젝트의 표시 여부 및 표현 방식 정보

□ REMOVE OBJECT

오브젝트를 삭제한다.



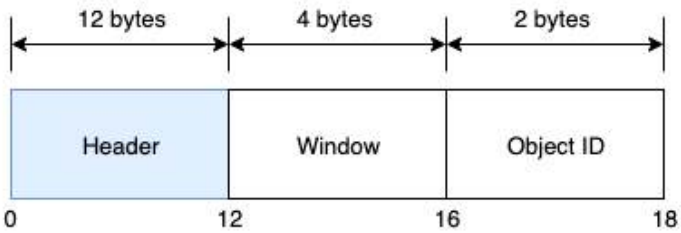
<figure 29> REMOVE OBJECT request octet string

<table 21> REMOVE OBJECT request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 오브젝트가 속한 윈도우
Object ID	*OID*	2	• 오브젝트 아이디

□ SELECT OBJECT

오브젝트를 선택한다.



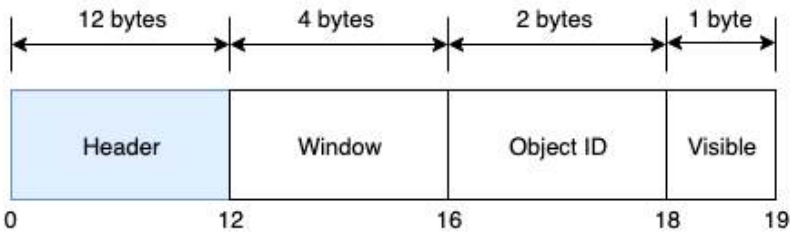
<figure 30> SELECT OBJECT request octet string

<table 22> SELECT OBJECT request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 선택한 오브젝트가 속한 윈도우
Object ID	*OID*	2	• 오브젝트 아이디

□ SET OBJECT VISIBLE

오브젝트의 표시 여부를 변경한다.



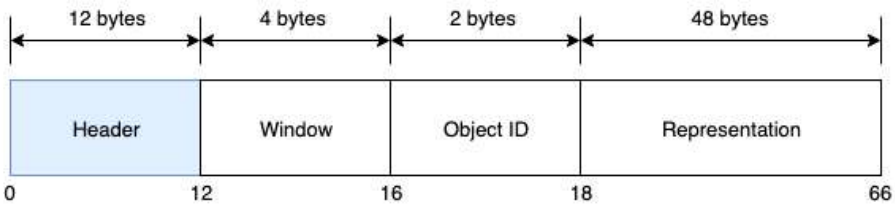
<figure 31> SET OBJECT VISIBLE request octet string

<table 23> SET OBJECT VISIBLE request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 오브젝트가 속한 윈도우
Object ID	*OID*	2	• 오브젝트 아이디
Visible	bool	1	• 표시 여부

□ SET OBJECT REPRESENTATION

오브젝트의 표현 방식을 변경한다.



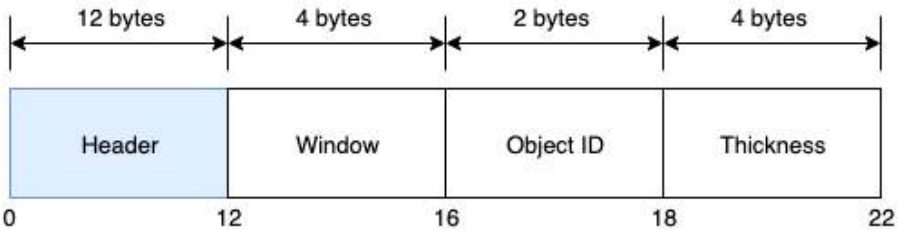
<figure 32> SET OBJECT REPRESENTATION request octet string

<table 24> SET OBJECT REPRESENTATION request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 오브젝트가 속한 윈도우
Object ID	*OID*	2	• 오브젝트 아이디
Representation	*OBJREP*	48	• 오브젝트의 표현 방식 정보

□ SET CONTOURLINE THICKNESS

Contour 라인의 두께를 변경한다.



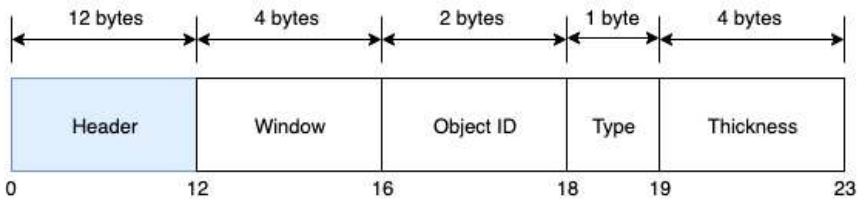
<figure 33> SET CONTOURLINE THICKNESS request octet string

<table 25> SET CONTOURLINE THICKNESS request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 오브젝트가 속한 윈도우
Object ID	*OID*	2	• 오브젝트 아이디
Thickness	float	4	• 두께

□ SET LINE REPRESENTATION

라인 오브젝트의 표현 방식을 변경한다.



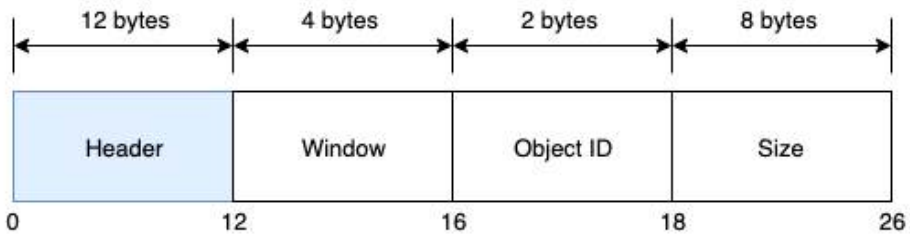
<figure 34> SET LINE REPRESENTATION request octet string

<table 26> SET LINE REPRESENTATION request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 오브젝트가 속한 윈도우
Object ID	*OID*	2	• 오브젝트 아이디
Type	uint8_t	1	• 라인 종류(Line, Ribbon, Tube)
Thickness	float	4	• 두께

□ SET GLYPH SIZE

글리프(Glyph) 오브젝트의 크기를 변경한다.



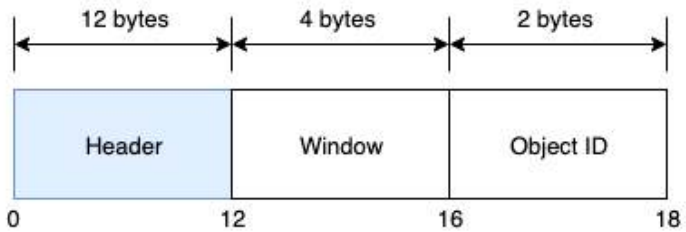
<figure 35> SET GLYPH SIZE request octet string

<table 27> SET GLYPH SIZE request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 오브젝트가 속한 윈도우
Object ID	*OID*	2	• 오브젝트 아이디
Size	double	8	• 글리프(Glyph) 크기

□ START PARTICLE ANIMATION

라인 오브젝트의 파티클 애니메이션을 시작한다.



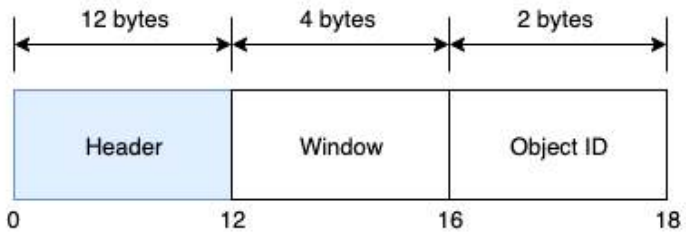
<figure 36> START PARTICLE ANIMATION request octet string

<table 28> START PARTICLE ANIMATION request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 파티클 애니메이션을 시작할 오브젝트가 속한 윈도우
Object ID	*OID*	2	• 오브젝트 아이디

□ STOP PARTICLE ANIMATION

라인 오브젝트의 파티클 애니메이션을 종료한다.



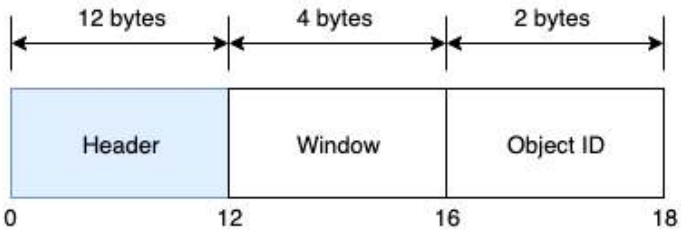
<figure 37> STOP PARTICLE ANIMATION request octet string

<table 29> STOP PARTICLE ANIMATION request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 파티클 애니메이션을 종료할 오브젝트가 속한 윈도우
Object ID	*OID*	2	• 오브젝트 아이디

□ START SLICE ANIMATION

멀티 슬라이스 오브젝트의 슬라이스 애니메이션을 시작한다.



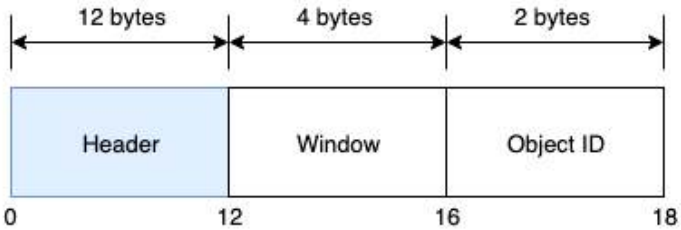
<figure 38> START SLICE ANIMATION request octet string

<table 30> START SLICE ANIMATION request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 오브젝트가 속한 윈도우
Object ID	*OID*	2	• 오브젝트 아이디

□ STOP SLICE ANIMATION

멀티 슬라이스 오브젝트의 슬라이스 애니메이션을 종료한다.



<figure 39> STOP SLICE ANIMATION request octet string

<table 31> STOP SLICE ANIMATION request specification

필드	타입	바이트	설명
Window	*WKEY*	4	• 오브젝트가 속한 윈도우
Object ID	*OID*	2	• 오브젝트 아이디

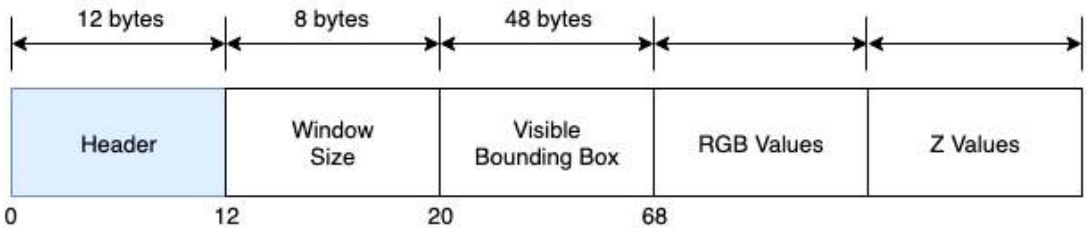
6. 응답 메시지 상세

본 절에서는 응답 메시지의 상세 내용에 대해 알아본다. 응답 메시지는 크게 이미지, 상태로 구분된다. 각 메시지의 명세는 필드, 타입, 바이트, 설명으로 구성된 표로 설명하며, 프로토콜에서 정의한 데이터 타입은 이탤릭체로 표기되며 상세 내용은 Predefined Types 항목을 참조한다.

가. 이미지

○ IMAGE

특정 윈도우의 현재 이미지를 표현한다.



<figure 40> IMAGE response octet string

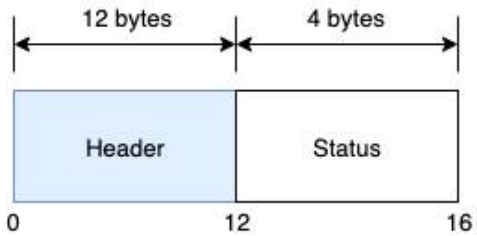
<table 32> IMAGE response specification

필드	타입	바이트	설명
Window Size	uint16 [2]	8	• 윈도우의 크기(너비, 높이)
Visible Bounding Box	double [6]	48	• 윈도우에서 현재 카메라에 보이고 있는 바운딩박스 정보(minX, maxX, minY, maxY, minZ, maxZ)
RGB Values	double [#pixels * 3]	-	• 이미지의 RGB 정보
Z Values	float [#pixels]	-	• 이미지의 Z 정보

나. 상태

○ STATUS

RENDER를 제외한 모든 요청의 응답 메시지로 사용된다. 요청에 대한 결과가 특정 메시지가 아니라 단순한 성공, 실패일 때 이를 전달한다.



<figure 41> STATUS response octet string

<table 33> STATUS response specification

필드	타입	바이트	설명
Status	uint32_t	4	• 실행 결과(0이면 성공, 아니면 실패)

7. Predefined Types

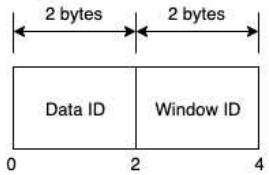
가. Primitive 타입

<table 34> Primitive types

<table>
<tr><th>필드</th><th>타입</th><th>바이트</th><th>설명</th></tr>
<tr><td>DID</td><td>int16_t</td><td>2</td><td>• 데이터 셋 아이디</td></tr>
<tr><td>WID</td><td>int16_t</td><td>2</td><td>• 윈도우 아이디</td></tr>
<tr><td>EID</td><td>int16_t</td><td>2</td><td>• Element 아이디</td></tr>
<tr><td>VID</td><td>int16_t</td><td>2</td><td>• Variable 아이디</td></tr>
<tr><td>VIDX</td><td>int16_t</td><td>2</td><td>• 벡터(vector) 또는 텐서(tensor) 변수의 서브 컴포넌트의 인덱스
• 벡터는 0~2, 텐서는 0~8</td></tr>
<tr><td>TS</td><td>int16_t</td><td>2</td><td>• 타임스텝</td></tr>
<tr><td>BID</td><td>int16_t</td><td>2</td><td>• 블록 아이디</td></tr>
<tr><td>DF</td><td>uint8_t</td><td>1</td><td>• 데이터 포맷
<table>
<tr><th>구분</th><th>코드</th></tr>
<tr><td>GLOVE</td><td>0x01</td></tr>
<tr><td>CGNS</td><td>0x02</td></tr>
<tr><td>ENSIGHT</td><td>0x03</td></tr>
<tr><td>FLUENT</td><td>0x04</td></tr>
<tr><td>OPENFOAM</td><td>0x05</td></tr>
<tr><td>PLOT3D</td><td>0x06</td></tr>
<tr><td>TECPLOT</td><td>0x07</td></tr>
<tr><td>VTK</td><td>0x08</td></tr>
</table></td></tr>
<tr><td>GT</td><td>uint8_t</td><td>1</td><td>• 격자 종류
<table>
<tr><th>구분</th><th>코드</th></tr>
<tr><td>정렬격자</td><td>0x03</td></tr>
<tr><td>비정렬격자</td><td>0x04</td></tr>
</table></td></tr>
<tr><td>OID</td><td>int16_t</td><td>2</td><td>• 오브젝트 아이디</td></tr>
<tr><td>TID</td><td>int16_t</td><td>2</td><td>• 트랜잭션 아이디</td></tr>
</table>

나. WKEY(Window Key)

렌더링 요청 및 응답에서 특정 윈도우를 지정하기 위해서는 (데이터 셋 ID, 윈도우 ID) 조합이 필요하다. 사용하지 않는 경우 (-1, -1)로 초기화한다.



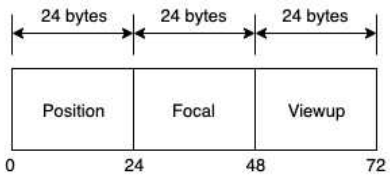
<figure 42> WKEY octet string

<table 35> WKEY specification

필드	타입	바이트	설명
Data ID	*DID*	2	• 데이터 셋 아이디
Window ID	*WID*	2	• 윈도우 아이디

다. CAMERA

카메라 정보를 표현한다. 카메라의 위치, 초점 위치, 상향 벡터로 구성된다.



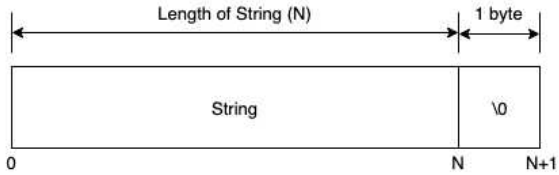
<figure 43> CAMERA octet string

<table 36> CAMERA specification

필드	타입	바이트	설명
Position	double [3]	24	• 카메라의 위치(x, y, z)
Focal	double [3]	24	• 카메라의 초점 위치(x, y, z)
Viewup	double [3]	24	• 카메라의 상향 벡터(x, y, z)

라. STRING

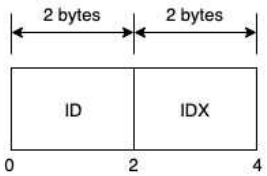
별도의 문자열 길이 정보 없이 문자열 뒤에 따라오는 '\0' 으로 문자열의 끝을 확인한다. 실제 길이는 (문자열 길이 + 1) 바이트이다.



<figure 44> STRING octet string

마. VKEY(Variable Key)

가시화 요청 및 응답에서 데이터 셋의 특정 변수를 지정하기 위해서는 (변수 ID, 서브 컴포넌트의 인덱스) 조합이 필요하다. 인덱스가 가질 수 있는 값을 살펴보면, 스칼라 변수는 서브 컴포넌트가 없으므로 -1, 벡터 변수는 3개의 서브 컴포넌트를 가지므로 0~2, 텐서 변수는 9개의 서브 컴포넌트를 가지므로 0~8이다. 그리고 벡터 또는 텐서 변수의 전체를 지정하려면 (변수 ID, -1)을 사용한다. 그리고 사용하지 않는 경우 (-1, -1)로 초기화한다.



<figure 45> VKEY octet string

<table 37> VKEY specification

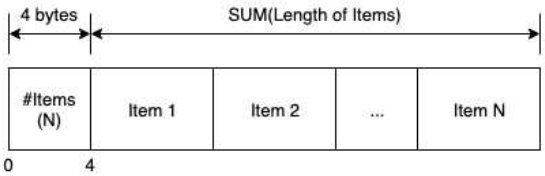
필드	타입	바이트	설명
ID	*VID*	2	• Variable 아이디
IDX	*VIDX*	2	• 벡터(vector) 또는 텐서(tensor) 변수의 서브 컴포넌트의 인덱스

<table 38> VKEY usage example

변수 타입	VKEY	설명
스칼라	(2, -1)	• 2번 변수(스칼라)
벡터	(3, -1)	• 3번 변수(벡터) 전체
	(3, 1)	• 3번 변수(벡터)의 1번 컴포넌트
텐서	(4, -1)	• 4번 변수(텐서) 전체
	(4, 7)	• 4번 변수(텐서)의 7번 컴포넌트

바. LIST(Item)

아이템의 리스트를 표현한다. 아이템 개수 뒤에 각 아이템의 정보가 순서대로 이어진다. 각 아이템의 길이는 서로 다를 수 있다.



<figure 46> LIST octet string

<table 39> LIST specification

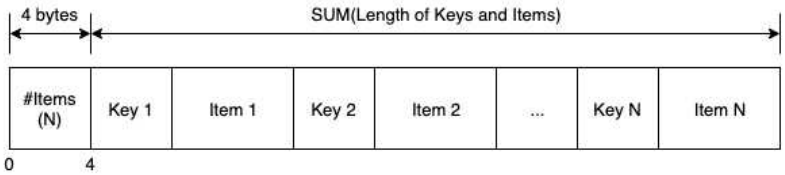
필드	타입	바이트	설명
#Items	int32_t	4	• 아이템의 개수
Item	-	-	• 각 아이템의 길이는 서로 다를 수 있음

사. MAP(Key, Item)

(Key, Item)으로 구성된 맵을 표현한다. 아이템의 개수 뒤에 Key, 아이템의 정보가 순서대로 이어진다. Key의 길이는 모두 동일하지만, 각 아이템의 길이는 서로 다를 수 있다.

<table 40> MAP specification

필드	타입	바이트	설명
#Items	uint32_t	4	• 아이템의 개수
Key	-	-	• Key의 길이는 모두 동일
Item	-	-	• 각 아이템의 길이는 서로 다를 수 있음



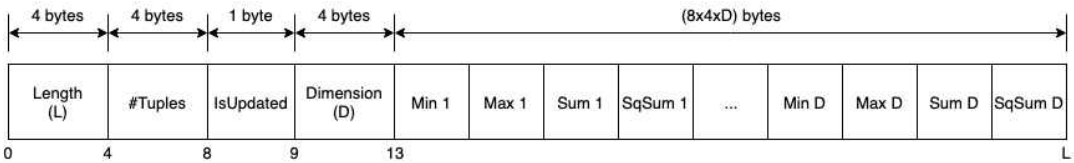
<figure 47> MAP octet string

아. METADATA

데이터 셋에 대한 필요한 모든 정보를 표현하기 때문에 복잡한 구조로 구성된다. 먼저 내부적으로 사용되는 타입을 설명한 후, 메타데이터 전체 정보를 설명한다. 그리고 GDM 내부에서 사용하지만 클라이언트에서 필요 없는 정보에 대한 설명은 생략한다.

□ AGGR

변수의 통계 정보(최대, 최소, 합계, 평균)를 표현한다.



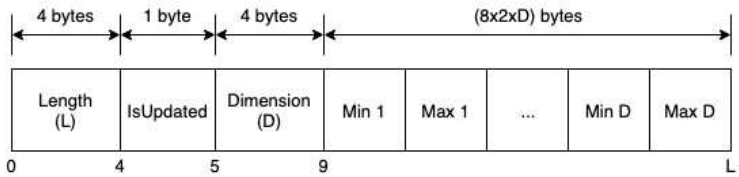
<figure 48> AGGR octet string

<table 41> AGGR specification

필드	타입	바이트	설명
Length	uint32_t	4	• AGGR 바이너리의 길이(Length 포함)
#Tuples	uint32_t	4	• 데이터 튜플의 개수(평균 계산에 활용)
IsUpdated	bool	1	• GDM에서 사용하는 내부용 변수
Dimension	int32_t	4	• 변수의 Dimension(D)
Array	double [4*D]	8*4*D	• Min, Max, Sum, Squared Sum 순서로 반복

□ BBOX

격자의 경계 정보를 표현한다.



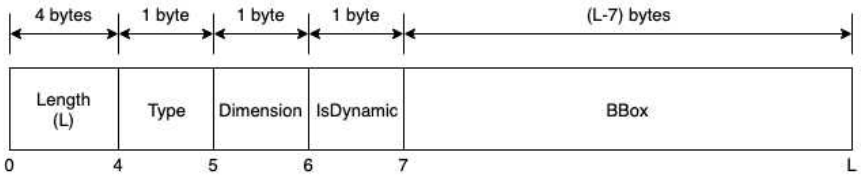
<figure 49> BBOX octet string

<table 42> BBOX specification

필드	타입	바이트	설명
Length	uint32_t	4	• BBOX 바이너리의 길이(Length 포함)
IsUpdated	bool	1	• GDM에서 사용하는 내부용 변수
Dimension	int32_t	4	• 변수의 Dimension(D)
Array	double [2*D]	8*2*D	• Min, Max 순서로 반복

□ MESH

격자 정보를 표현한다.



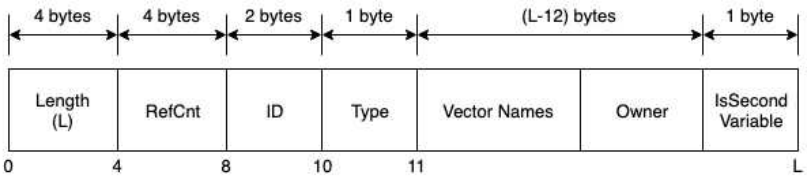
<figure 50> MESH octet string

<table 43> MESH specification

필드	타입	바이트	설명
Length	uint32_t	4	• MESH 바이너리의 길이(Length 포함)
Type	uint8_t	1	• 격자 데이터 타입(float, double) 구분 / 코드 FLOAT / 0x09 DOUBLE / 0x0A
Dimension	uint8_t	1	• 격자의 Dimension
IsDynamic	bool	1	• 격자의 시변환 여부
BBox	*MAP(TS, BBOX)*	-	• 타임스텝별 격자의 경계 정보

□ VAR

변수 정보를 표현한다.



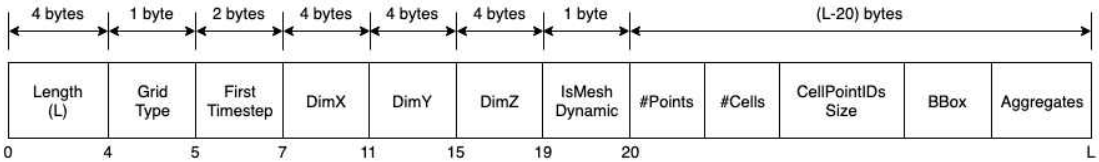
<figure 51> VAR octet string

<table 44> VAR specification

필드	타입	바이트	설명
Length	uint32_t	4	• VAR 바이너리의 길이(Length 포함)
RefCnt	int32_t	4	• GDM에서 사용하는 내부용 변수
ID	*VID*	2	• Variable 아이디
Type	uint8_t	1	• 격자 데이터 타입(int, float, double) 구분 / 코드 INT / 0x05 FLOAT / 0x09 DOUBLE / 0x0A
Vector Names	*MAP*(uint8_t, *STRING*)	-	• 벡터 또는 텐서 변수의 서브 컴포넌트 이름
Owner	*STRING*	-	• GDM에서 사용하는 내부용 변수
IsSecondVariable	bool	1	• 2차변수 여부

□ BLOCK

Element를 구성하는 블록의 정보를 표현한다.



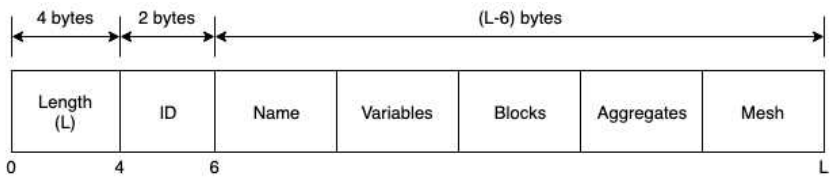
<figure 52> BLOCK octet string

<table 45> BLOCK specification

필드	타입	바이트	설명
Length	uint32_t	4	• BLOCK 바이너리의 길이(Length 포함)
Grid Type	*GT*	1	• 격자 종류
FirstTimestep	*TS*	2	• GDM에서 사용하는 내부용 변수
DimX	int32_t	4	• 정렬격자 I 정보
DimY	int32_t	4	• 정렬격자 J 정보
DimZ	int32_t	4	• 정렬격자 K 정보
IsMeshDynamic	bool	1	• 격자의 시변환 여부
#Points	*MAP(TS*, int32_t)	-	• 타임스텝별 포인트 개수
#Cells	*MAP(TS*, int32_t)	-	• 타임스텝별 셀 개수
CellPointIDsSize	*MAP(TS*, int32_t)	-	• 타임스텝별 셀 포인트 정보 개수
BBox	*MAP(TS*, int32_t)	-	• 타임스텝별 격자의 경계 정보
Aggregates	*MAP(VID*, *MAP(TS*, *AGGR*))	-	• 변수별 타임스텝별 통계 정보

□ ELEM

데이터를 구성하는 Element 정보를 표현한다.



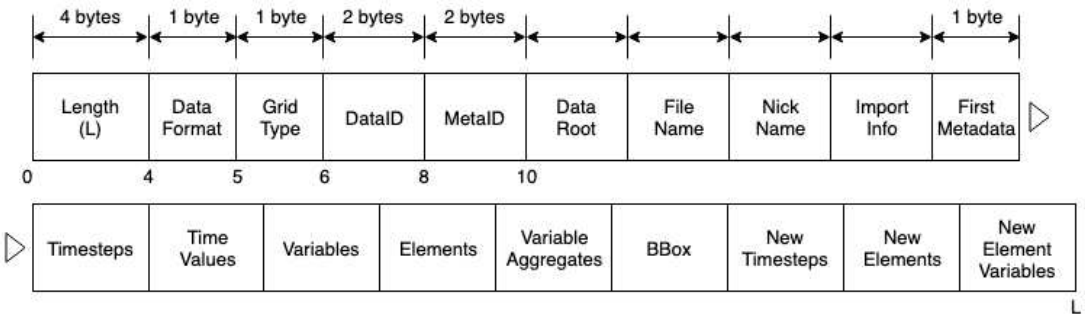
<figure 53> ELEM octet string

<table 46> ELEM specification

필드	타입	바이트	설명
Length	uint32_t	4	• ELEM 바이너리의 길이(Length 포함)
ID	*EID*	2	• Element 아이디
Name	*STRING*	-	• Element 이름
Variables	*LIST(VID)*	-	• Variable 아이디 목록
Blocks	*LIST(BLOCK)*	-	• 블록 정보
Aggregates	*MAP(VID, MAP(TS, AGGR))*	-	• 변수별 타임스텝별 통계 정보
Mesh	*MESH*	-	• 격자 정보

□ METADATA

메타데이터는 다음의 정보로 구성된다.



<figure 54> METADATA octet string

<table 47> METADATA specification

필드	타입	바이트	설명
Length	uint32_t	4	• ELEM 바이너리의 길이(Length 포함)
Data Format	*DF*	1	• 데이터 포맷
Grid Type	*GT*	1	• 격자 종류
Data ID	*DID*	2	• 서버에 로드된 데이터 셋의 아이디
Meta ID	*DID*	2	• GDM에서 사용하는 내부용 변수
Data Root	*STRING*	-	• GDM에서 사용하는 내부용 변수
File Name	*STRING*	-	• 메타파일 경로
Nick Name	*STRING*	-	• 데이터 셋의 별칭
Import Info	*STRING*	-	• Import 정보
FirstMetadata	bool	1	• GDM에서 사용하는 내부용 변수
Timesteps	*LIST(TS)*	-	• 타임스텝 정보
Time Values	*LIST*(double)	-	• 실제 시간 정보
Variables	*LIST(VAR)*	-	• Variable 정보
Elements	*LIST(ELEM)*	-	• Element 정보
Variable Aggregates	*MAP(VID, MAP(TS, AGGR))*	-	• 변수별 타임스텝별 통계 정보
BBox	*MAP(TS, BBOX)*	-	• 타임스텝별 격자의 경계 정보
New Timesteps	*LIST(TS)*	-	• GDM에서 사용하는 내부용 변수
New Elements	*LIST(EID)*	-	• GDM에서 사용하는 내부용 변수
New Element Variables	*MAP(EID, LIST(VID))*	-	• GDM에서 사용하는 내부용 변수

자. DCTX

데이터 셋의 컨텍스트에 대한 필요한 모든 정보를 표현하기 때문에 복잡한 구조로 구성된다. 먼저 내부적으로 사용되는 타입을 설명한 후, 컨텍스트 전체 정보를 설명한다.

□ REQ

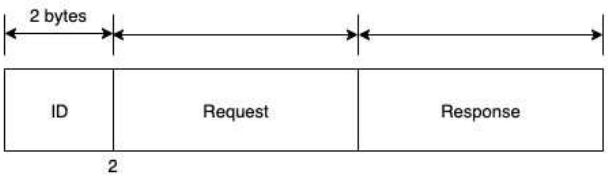
가시화 오브젝트를 생성하기 위한 가시화 요청을 표현한다. 내용이 방대한 관계로 요청 종류별 상세 정보는 여기에서 설명하지 않는다.

□ RES

가시화 요청에 대한 서버의 결과 데이터를 표현한다. 내용이 방대한 관계로 응답 종류별 상세 정보는 여기에서 설명하지 않는다.

□ TR(Transaction)

가시화 오브젝트를 관리하기 위해서는 가시화 요청과 이에 대한 결과 데이터가 필수적이기 때문에 이를 트랜잭션이라 정의하고, 트랜잭션을 기본 단위로 관리한다.



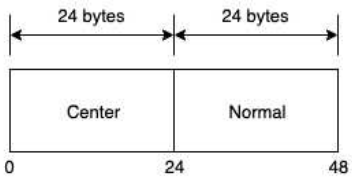
<figure 55> TR octet string

<table 48> TR specification

필드	타입	바이트	설명
ID	*TID*	2	• 트랜잭션 아이디
Request	*REQ*	-	• 가시화 요청 정보
Response	*RES*	-	• 가시화 결과 데이터

□ MIRROR

윈도우에 대한 미러링 정보를 표현한다.



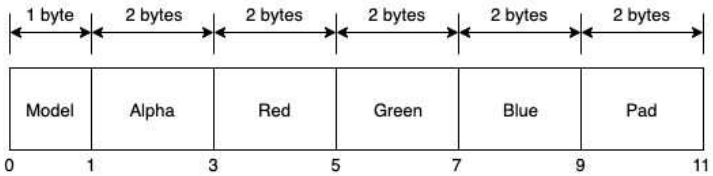
<figure 56> MIRROR octet string

<table 49> MIRROR specification

필드	타입	바이트	설명
Center	double [3]	24	• 미러링 평면의 중심(x, y, z)
Normal	double [3]	24	• 미러링 평면의 노말 벡터(x, y, z)

□ COLOR

색상 정보를 표현한다.



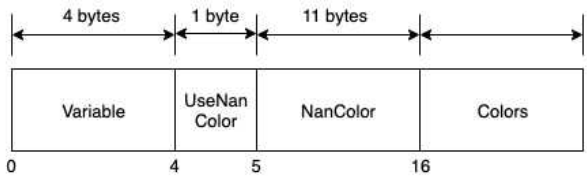
<figure 57> COLOR octet string

<table 50> COLOR specification

필드	타입	바이트	설명
Model	int8_t	1	• 색상 모델(RGB, HSV, CMYK, HSL)
Alpha	uint16_t	2	• 색상의 alpha 값
Red	uint16_t	2	• 색상의 red 값
Green	uint16_t	2	• 색상의 green 값
Blue	uint16_t	2	• 색상의 blue 값
Pad	uint16_t	2	• 색상 모델이 변경되더라도 고정 길이를 유지하기 위한 용도

□ COLORMAP

특정 변수에 대한 컬러맵을 표현한다.



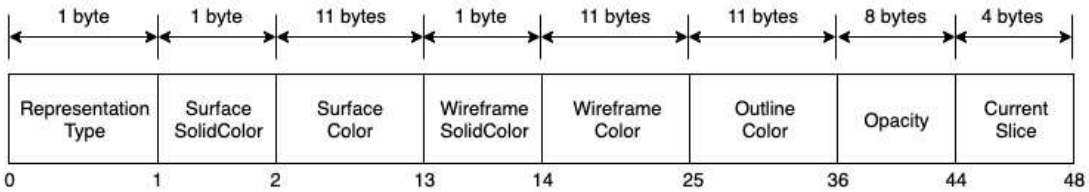
<figure 58> COLORMAP octet string

<table 51> COLORMAP specification

필드	타입	바이트	설명
Variable	*VKEY*	4	• 컬러맵을 적용할 변수
UseNanColor	bool	1	• NaN 색상 사용 여부
NanColor	*COLOR*	11	• NaN 색상
Colors	*MAP*(double, *COLOR*)	-	• 변수값에 따른 색상 정보

□ OBJREP(Object Representation)

오브젝트의 표현 방식을 표현한다. 오브젝트는 Surface, Wireframe, Outline의 세 부분으로 구성되며, 각 부분에 대한 색상 및 투명도를 설정한다. 그리고 멀티 슬라이스 오브젝트인 경우 현재 표시 중인 슬라이스의 인덱스 정보도 필요하다.



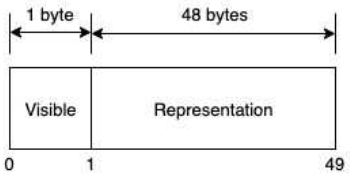
<figure 59> OBJREP octet string

<table 52> OBJREP specification

필드	타입	바이트	설명
Representation Type	uint8_t	1	• Surface, Wireframe, Outline의 표시 여부 구분 / 코드 None / 0x00 Surface / 0x01 Wireframe / 0x02 Outline / 0x04
Surface SolidColor	bool	1	• Surface 색상을 지정하는지 여부
SurfaceColor	*COLOR*	11	• Surface의 색상
Wireframe SolidColor	bool	1	• Wireframe 색상을 지정하는지 여부
WireframeColor	*COLOR*	11	• Wireframe의 색상
OutlineColor	*COLOR*	11	• Outline의 색상
Opacity	double	8	• 투명도(0~1)
CurrentSlice	int32_t	4	• 멀티 슬라이스 오브젝트인 경우 현재 표시 중인 슬라이스의 인덱스 • 사용하지 않으면 -1로 설정

□ OBJ

오브젝트의 화면 표시 여부와 표현 방식을 함께 포함한다.



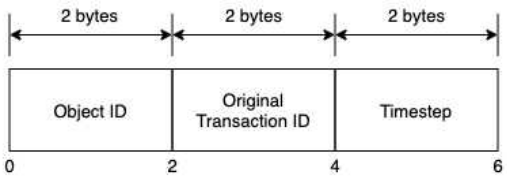
<figure 60> OBJ octet string

<table 53> OBJ specification

필드	타입	바이트	설명
Visible	bool	1	• 표시 여부
Representation	*OBJREP*	48	• 오브젝트의 표현 방식 정보

□ TKEY

가시화 결과 데이터는 크기가 크기 때문에 효율적인 메모리 활용을 위해 중복 없이 관리하는 것이 중요하다. 이를 위해 트랜잭션을 (오브젝트, 원본 트랜잭션 아이디, 타임스텝)을 Key로 관리한다.



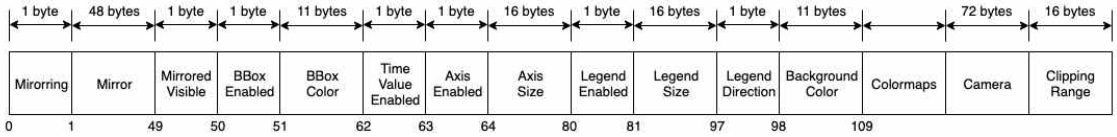
<figure 61> TKEY octet string

<table 54> TKEY specification

필드	타입	바이트	설명
Object ID	*OID*	2	• 오브젝트 아이디
Original Transaction ID	*TID*	2	• 결과 데이터를 공유하고 있는 원본 트랜잭션 아이디
Timestep	*TS*	2	• 타임스텝

□ WINDOW

하나의 윈도우를 표현한다.



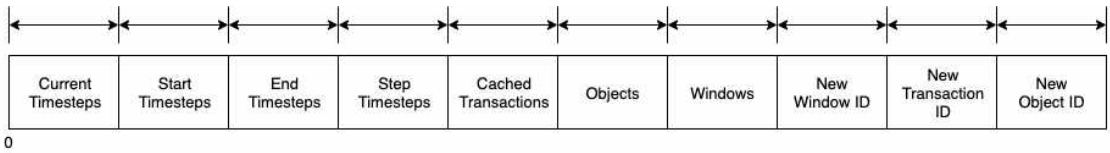
<figure 62> WINDOW octet string

<table 55> WINDOW specification

필드	타입	바이트	설명
Mirroring	bool	1	• 미러링 적용 여부
Mirror	*MIRROR*	48	• 미러링 평면 정보
Mirrored Visible	bool	1	• 미러링 오브젝트 화면 표시 여부
BBox Enabled	bool	1	• 바운딩박스 화면 표시 여부
BBox Color	*COLOR*	11	• 바운딩박스의 색상
TimeValue Enabled	bool	1	• 실제 시간 화면 표시 여부
Axis Enabled	bool	1	• 축 화면 표시 여부
Axis Size	double [2]	16	• 축 사이즈 정보(w, h)
Legend Enabled	bool	1	• 컬러 레전드 화면 표시 여부
Legend Size	double [2]	16	• 컬러 레전드 사이즈 정보(w, h)
Legend Direction	uint8_t	1	• 컬러 레전드 방향 구분 / 코드 TOP / 0x00 BOTTOM / 0x01 LEFT / 0x02 RIGHT / 0x03
Background Color	*COLOR*	11	• 배경화면의 색상
Colormaps	*MAP(VKEY, COLORMAP)*	-	• 변수별 컬러맵 정보
Camera	*CAMERA*	72	• 카메라 정보
Clipping Range	double [2]	16	• 카메라 클리핑 범위 정보

□ DCTX

데이터 컨텍스트는 다음의 정보로 구성된다.



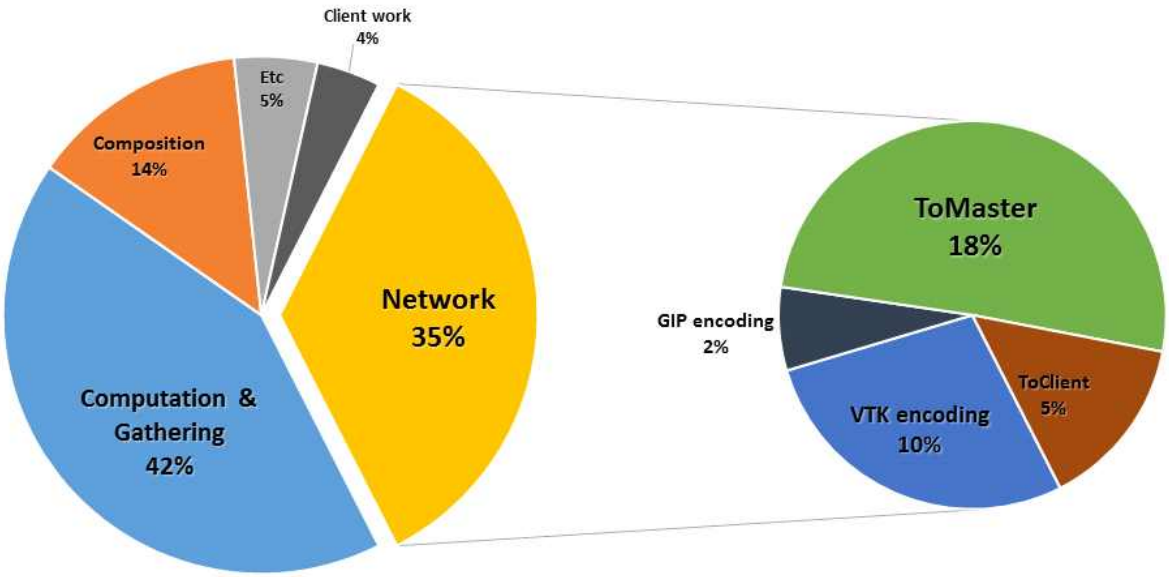
<figure 63> DCTX octet string

<table 56> DCTX specification

필드	타입	바이트	설명
Current Timesteps	*MAP(WID, TS)*	-	• 윈도우별 현재 타임스텝 정보
Start Timesteps	*MAP(WID, TS)*	-	• 윈도우별 첫 타임스텝 정보
End Timesteps	*MAP(WID, TS)*	-	• 윈도우별 마지막 타임스텝 정보
Step Timesteps	*MAP(WID, TS)*	-	• 윈도우별 타임스텝 이동간격 정보
Cached Transactions	*MAP(TKEY, TR)*	-	• 트랜잭션 정보
Objects	*MAP(TKEY, OBJ)*	-	• 모든 오브젝트 정보
Windows	*MAP(WID, WINDOW)*	-	• 컨텍스트를 구성하는 모든 윈도우 정보
New Window ID	*WID*	2	• 새로 발급할 윈도우 아이디
New Transaction ID	*TID*	2	• 새로 발급할 트랜잭션 아이디
New Object ID	*OID*	2	• 새로 발급할 오브젝트 아이디

8. 결론

거대 컴퓨팅 데이터의 분석은 시간 소모적인 일이다. 데이터가 클수록 훨씬 더 많은 가시화 시간이 필요하다. 가시화에 소요되는 시간을 분석하면, 분산병렬 가시화 시스템의 경우 네트워크 소요시간이 사용자 요청부터 응답까지의 시간에 35%를 차지한다<figure 52>. 따라서 거대 컴퓨팅 가시화의 분산병렬 가시화에 있어 네트워크 부하를 줄이는 것은 전체 가시화 시간을 줄이는 데 반드시 필요한 일이다. 이미지 기반 가시화는 해상도가 정해지면 가시화 결과 데이터의 크기에 상관없이 동일한 데이터만 전송할 수 있다. 특히 저해상도 클라이언트의 경우 이 데이터 사이즈는 해상도만큼 줄일 수 있어 저 사양 단말 디바이스의 컴퓨팅 시간도 함께 줄일 수 있다.



<figure 64> GLOVE visualization system profiling result

본 연구에서 개발한 원격 제어 가시화 프로토콜의 경우, 사용자의 동작에 따른 빠른 응답 시간을 보장함으로써 좀 더 빠른 가시화 결과를 볼 수 있게 만드는 프로토콜이다. 향후, 초고해상도 이미지의 빠른 전송을 위한 압축 알고리즘 및 LOD (Level Of Detail)의 구현으로 최초 응답 시간을 더 단축하는 프로토콜도 반영할 예정이다.

비매품/무료
95560
9 788929 411565
ISBN 978-89-294-1156-5 (PDF)